Efficient solution of large-scale Riccati equations and an ODE framework for linear matrix equations

Von der

Carl-Friedrich-Gauß-Fakultät

der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines

Doktoringenieurs (Dr.-Ing.)

genehmigte

Dissertation

von

Christian Bertram

geboren am 10.08.1988

in Braunschweig

Eingereicht am: 28.04.2021

Disputation am: 01.09.2021

1. Referentin: Prof. Dr. Heike Faßbender

2. Referent: Prof. Dr. Zvonimir Bujanović

2021

Bertram, Christian

Efficient solution of large-scale Riccati equations and an ODE framework for linear matrix equations

Dissertation,

Technische Universität Carolo-Wilhelmina zu Braunschweig, 2021.

Zusammenfassung

Die Lösung von Matrixgleichungen wie Riccati-Gleichungen, Lyapunov-Gleichungen und Sylvester-Gleichungen wird in vielen Gebieten der angewandten Mathematik benötigt: Für einen linear-quadratischen Regler, der ein dynamisches System unter Berücksichtigung einer Optimalitätsbedingung in einen gewünschten Zustand überführt, ist die Lösung einer algebraischen Riccati-Gleichung nötig. In der Modellreduktion soll ein hochdimensionales dynamisches System durch ein kleineres dynamisches System mit möglichst ähnlichem Verhalten ersetzt werden. In linearen Systemen können hierzu weniger relevante Zustände vernachlässigt werden, wobei Relevanz durch die Energie eines Zustands bestimmt wird. Hierzu ist die Lösung von Lyapunov-Gleichungen erforderlich. Diese Arbeit befasst sich mit der numerischen Lösung hochdimensionaler Matrixgleichungen mittels iterativer Verfahren.

Aufgrund der Größe der algebraischen Riccati-Gleichung

$$0 = \mathcal{R}(X) := A^*X + XA + C^*C - XBB^*X$$

mit $A \in \mathbb{C}^{n\times n}$ groß und dünnbesetzt, $B \in \mathbb{C}^{n\times m}$, $C \in \mathbb{C}^{p\times n}$ kann die Lösung nicht direkt bestimmt werden, sondern wird durch eine approximative Lösung $\tilde{X} = ZYZ^*$ von geringem Rang angenähert. Hierbei wird Z als Basis eines rationalen Krylovraums gewählt und enthält nur wenige Spalten. Die innere Matrix Y_j ist klein und quadratisch. In dieser Arbeit werden zwei Wege untersucht, die Matrix Y zu wählen: Durch eine Rang-Bedingung an das Riccati-Residuum $\mathcal{R}(\tilde{X})$ und durch Projektion des Riccati-Residuums $\mathcal{R}(\tilde{X})$ auf den von Z erzeugten Krylovraum.

Die Rang-Bedingung wird durch die wohlbekannten ADI-Verfahren motiviert. Die approximativen ADI-Lösungen spannen einen Krylovraum auf und führen zu einem Riccati-Residuum vom Rang p. Es wird bewiesen, dass die Rang-p-Bedingung Existenz und Eindeutigkeit einer solchen approximativen Lösung impliziert. Aus diesem Ergebnis werden effiziente iterative Verfahren abgeleitet, die eine solche approximative Lösung erzeugen.

Bisher bekannte Projektionsverfahren werden auf schiefe Projektionen erweitert und es wird eine neue Formulierung des Riccati-Residuums hergeleitet, die eine effiziente Berechnung der Norm erlaubt. Weiter wird eine abgeschnittene approximative Lösung als Lösung einer Riccati-Gleichung charakterisiert, die auf einen Unterraum des von Z erzeugten Krylovraums projiziert wird.

Um die Lösung der Lyapunov-Gleichung

$$0 = \mathcal{L}(\mathcal{P}) \coloneqq A\mathcal{P} + \mathcal{P}A^{\mathsf{T}} + BB^{\mathsf{T}}$$

mit $A \in \mathbb{R}^{n\times n}$ und $B \in \mathbb{R}^{n\times m}$ zu approximieren wird ein System gewöhnlicher Differentialgleichungen mittels Runge-Kutta-Verfahren numerisch gelöst. Es wird gezeigt, dass der von der approximativen Lösung aufgespannte Raum ein rationaler Krylovraum ist, dessen Pole von den Zeitschrittweiten der Integration und den Eigenwerten der Koeffizientenmatrix aus dem Butcher-Tableau des verwendeten Runge-Kutta-Verfahrens abhängen. Das Verfahren wird auf ein Problem der Modellreduktion angewendet.

Die analytische Lösung des Differentialgleichungssystems erfüllt eine algebraische Invariante. Diejenigen Runge-Kutta-Verfahren, die diese Invariante erhalten, werden durch eine Bedingung an die zugehörigen Butcher-Tableaus charakterisiert. Es wird gezeigt, dass diese speziellen Verfahren äquivalent zur ADI-Iteration sind. Die Charakterisierung der ADI-Iteration als numerische Integration wird genutzt um die Äquivalenz eines weiteren, auf Legendre-Polynomen basierenden, Ansatzes zu zeigen. Der Invarianten-Ansatz wird auf Sylvester-Gleichungen übertragen.

Acknowledgements

At first I want to thank Heike Faßbender for supervising this thesis and for her guidance during the last years. She has made so many things possible and her comments clearly helped to improve this work. I would also like to thank Zvonimir Bujanović for being the second referee of my thesis and for his interesting questions. I further thank Jens-Peter Kreiß and Michael Herrmann for being members of my doctoral committee.

Special thanks goes to my Agnumerik colleagues Anna Bertram, Matthias Bollhöfer, Philip Saltenberger, Tanja Schenk, Michel-Niklas Senn, Nikta Shayanfar, Lena Vestveber and Tom Werner. I enjoyed working with you and will miss our conversations and discussions. I want to point out Anna and Philip, who always cheered me up in stressful times and proofread this work.

I thank Jens Saak, Patrick Kürschner and once again Zvonimir Bujanović for the discussions about shift strategies and for sharing their RADI implementation with me. I am also grateful for the introduction into the M.E.S.S. development and for the possibility to contribute my results.

Abschließend möchte ich meiner Familie danken, die mich immer unterstützt hat. Ohne die Förderung meiner Eltern schon zu Schulzeiten wäre diese Arbeit nicht möglich gewesen. Ich bin außerdem dankbar für die vielen Spaziergänge mit Lotte, die stets zu neuen Ideen und Ansätzen geführt haben.

Publications

Parts of this work have already been published. The Riccati ADI sections in the second part are based on

> Christian Bertram and Heike Faßbender. "Riccati ADI: Existence, uniqueness and new iterative methods". In: *arXiv e-prints* (2020). arXiv: 2004.11212 [math.NA].

The ODE framework introduced in the third part is based on the publications

> Christian Bertram and Heike Faßbender. "A quadrature framework for solving Lyapunov and Sylvester equations". In: *Linear Algebra and its Applications* 622 (2021), pp. 66–103. DOI: https://doi.org/10.1016/j.laa.2021.03.029

and

> Christian Bertram and Heike Faßbender. "A Link Between Gramian-Based Model Order Reduction and Moment Matching". In: *Model Reduction of Complex Dynamical Systems*. Ed. by Peter Benner et al. Cham: Springer International Publishing, 2021, pp. 119–139. ISBN: 978-3-030-72983-7. DOI: 10.1007/978-3-030-72983-7_6.

Contents

List of Figures

List of Tables

List of Algorithms

List of Symbols

Abbreviations

ADI	alternating direction implicit
BRAD	block rational Arnoldi decomposition
DIRK	diagonally implicit Runge-Kutta
EKSM	extendend Krylov subspace method
GPM	general projection method
LTI	linear time invariant
MIMO	multiple-input multiple-output
ODE	ordinary differential equation
R^2ADi	Riccati RAD iteration
RAD	rational Arnoldi decomposition
RADI	Riccati ADI
RKD	rational Krylov decomposition
RKSM	rational Krylov subspace method
SMW	Sherman-Morrison-Woodbury

Mathematical symbols and notation

$\deg(\mathfrak{p})$	degree of the polynomial $\mathfrak{p}$
$\operatorname{diag}(A_1, \ldots, A_l)$	(block) diagonal matrix with diagonal entries $A_1, \ldots, A_l$
$\imath$	imaginary unit with $\imath^2 = -1$
$\lambda(A)$	spectrum of A
$\mathcal{K}_j(A, b)$	polynomial Krylov subspace of order j
$\mathcal{K}_j(A, b, \mathfrak{q})$	rational Krylov subspace
$\mathcal{K}_j^+(A, b, \mathfrak{q})$	augmented Krylov subspace
$\mathcal{K}_j^\square(A, \boldsymbol{b}, \mathfrak{q})$	block Krylov subspace
$\mathbb{1}_s$	the s-dimensional vector $[1, \ldots, 1]^\mathsf{T}$ containing only ones

Π	projection matrix
$\mathbf{\Pi}_{(j)}$	set of all polynomial with degree at most j
$\underline{K_j}$	a matrix with more rows than columns
$\underline{K_{-j}}$	square lower submatrix of $\underline{K_j}$
e_i	the ith canonical standard basis vector
I_n	identity matrix of dimension n
$X \otimes Y$	Kronecker product of matrices X and Y
A^{T}	transposed matrix
A^*	complex conjugated of the transposed matrix A^{T}
$\mathbb{N}$	set of natural number
$\mathbb{N}_0$	set of natural numbers and zero
$\mathbb{R}$	set of real numbers
$\mathbb{R}_-$	set of negative real numbers
$\mathbb{R}_+$	set of positive real numbers
$\mathbb{C}$	set of complex numbers
$\mathbb{C}_-$	set of complex numbers with negative real part
$\mathbb{C}_+$	set of complex numbers with positive real part
$\overline{\mathbb{C}}$	extended complex numbers $\mathbb{C} \cup \{\infty\}$

Part I.

Preliminary results

1. Motivation

Matrix equations, such as Riccati and Lyapunov equations, appear in numerous fields of applied mathematics, scientific computing or in engineering problems. Their solution is required in e.g. linear-quadratic regulator problems of optimal control, in $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control, in balancing-related model order reduction or in Kalman filtering, see, e.g. [50, 23, 3, 27, 32]. The goal of optimal control is to steer a dynamical system to a wanted state in a way such that an optimality condition is satisfied. In linear-quadratic control for this the solution of a Riccati equation is required. In the field of model order reduction one would like to replace large-scale dynamical systems by smaller ones which preserve the behavior of the system. One way to accomplish this is to keep only the important states of the system and truncate the rest. In linear systems importance can be measured by the energy corresponding to a state, which requires the solution of Lyapunov equations.

With increasing computing power model dimensions became larger and larger over the last decades, making classical direct algorithms computationally unfeasible. Due to the large size it is challenging if not impossible to even store the solution: Assume that one entry in a matrix needs 8 bytes of memory, then a square matrix X with dimension 100 000 already needs more than 74.5 GB of memory. To make computations and storage feasible, the matrix X is approximated by a low rank matrix $\tilde{X} \approx X$. When the approximate solution $\tilde{X}$ has rank 200, then two 100 000-by-200 matrices Z_1 and Z_2 exist with $Z_1 Z_2^* = \tilde{X}$. These low rank factors need less than 300 MB for storage, reducing the necessary amount of memory by a factor of 250.

Numerous iterative methods have been developed to generate such low rank approximations by increasing the number of columns in the low rank factors until the desired accuracy is reached. Typically the approximation space, i.e. the space spanned by the

low rank factors, is a rational Krylov subspace. The columns are generated via the solution of shifted linear systems of equations, which is usually the most expensive part of the methods.

The matrix equations considered in this work are the continuous-time algebraic Riccati equation

$$\begin{aligned} 0 = \mathcal{R}(X) &:= A^*X + XA + C^*C - XBB^*X \\ &= (A^* - XBB^*)X + X(A - BB^*X) + C^*C + XBB^*X \end{aligned} \tag{1.1}$$

with complex matrices $A \in \mathbb{C}^{n\times n}$, $B \in \mathbb{C}^{n\times m}$ and $C \in \mathbb{C}^{p\times n}$, and the continuous time Lyapunov equation

$$0 = \mathcal{L}(\mathcal{P}) := A\mathcal{P} + \mathcal{P}A^{\mathsf{T}} + BB^{\mathsf{T}} \tag{1.2}$$

with real matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. We have a large and sparse system matrix A in mind and assume that the constant terms are of low rank, i.e. $p \ll n$, respectively $m \ll n$. With these assumptions the solution of the corresponding matrix equation has low numerical rank, which makes the approximate solution of (1.1) and (1.2) with low rank matrices possible [59, Sec. 3], [48, Ch. 2.3.3].

This thesis is divided into three parts. In the first part basic mathematical concepts are reviewed, which serve as foundation for the following parts. Rational Krylov subspaces are discussed, the evolution of the ADI iteration for linear matrix equations is outlined, basic knowledge about balancing based and interpolatory model order reduction is given and Runge-Kutta methods for the numerical solution of ordinary differential equations (ODEs) are presented.

The second part deals with the solution of large-scale algebraic Riccati equations and their low rank approximation using ADI-type and projection based methods. Rational Arnoldi decompositions are used to reformulate the Riccati residual. We characterize the approximate solution generated by the ADI-type methods through the rank of the residual in an existence and uniqueness result. It allows us to develop a novel, more

efficient and versatile ADI-type method. In projection methods the residual is projected to a (rational) Krylov subspace. We introduce an efficient projection method using rational Arnoldi decompositions which allows the use of oblique projections. A new characterization of truncated approximate solutions is given in terms of projections. In numerical experiments the effectiveness of the presented ADI and projection methods is shown.

In the third part methods for the solution of linear matrix equations based on numerical quadrature are presented. An ODE is derived whose solution converges against the solution of the Lyapunov equation for increasing time. Methods for its efficient numerical solution are introduced. A special family of Runge-Kutta methods which preserve a geometric invariant is identified and shown to be equivalent to the ADI iteration. For these particular Runge-Kutta methods the ODE approach is transferred to Sylvester equations.

2. Basic definitions and properties

We introduce basic definitions and properties which are used throughout this work.

2.1. Linear algebra

For $j \in \mathbb{N}_0$ a polynomial $\mathfrak{p}$ is given via $\mathfrak{p}(x) = \sum_{i=0}^{j} a_i x^i$ with coefficients $a_i \in \mathbb{C}$ for $i = 0, \ldots, j$. The *degree* $\deg(\mathfrak{p})$ of the polynomial $\mathfrak{p}$ is the highest index belonging to a nonzero coefficient. If all coefficients are zero, then the degree is set to $-\infty$. The set of all polynomials of degree at most j is denoted by $\mathbf{\Pi}_{(j)}$. If the highest coefficient is one then the polynomial is said to be *normalized*.

Let $\mathcal{V}$ be a linear subspace of $\mathbb{C}^n$ and $A \in \mathbb{C}^{n\times n}$. We call $\mathcal{V}$ *A-variant* if $A\mathcal{V} \not\subseteq \mathcal{V}$ and *A-invariant* otherwise. The space $\mathcal{V}^\perp = \{x \in \mathbb{C}^n \text{ with } x^*v = 0 \text{ for all } v \in \mathcal{V}\}$ is the *orthogonal complement* of $\mathcal{V}$. For a matrix V with $\operatorname{span}(V) = \mathcal{V}$ it holds $\mathcal{V}^\perp = \ker(V^*)$.

The following definitions are based on [35, 43]. Let A, B, $U \in \mathbb{C}^{n\times n}$. The matrix U is *unitary* if $U^*U = UU^* = I$ holds. If the matrix A satisfies $A = A^*$, then it is a *Hermitian*

matrix. A Hermitian matrix is *positive definite* if for all nonzero vectors $x \in \mathbb{C}^n$ it holds $0 < x^*Ax \in \mathbb{R}$, it is *positive semidefinite* if $0 \leq x^*Ax$ holds. The matrix A is *indefinite* if there exist vectors $x, y \in \mathbb{C}^n$ with $y^*Ay < 0 < x^*Ax$. The *Cholesky decomposition* of a Hermitian and positive semidefinite (positive definite) matrix A is given by the decomposition $A = R^*R$ with an upper triangular matrix $R \in \mathbb{C}^{n\times n}$ which has only nonnegative (positive) diagonal entries. For a positive semidefinite matrix A of rank j we also use the decomposition $A = ZZ^*$ with $Z \in \mathbb{C}^{n\times j}$ which we call *approximate Cholesky factorization*. Note that Z does not need to be upper triangular.

The set $\lambda(A, B) = \{z \in \mathbb{C} \colon \det(A - zB) = 0\}$ is the *spectrum* or *set of eigenvalues* of the *tuple* or *pencil* (A, B). The spectrum of A is given by $\lambda(A) = \lambda(A, I)$, i.e. it holds $\lambda(A) = \{z \in \mathbb{C} \colon \det(A - zI) = 0\}$. A nonzero vector $v \in \mathbb{C}^n$ is an *eigenvector* of (A, B) if $Av = \lambda Bv$ holds for some $\lambda \in \lambda(A, B)$. If $B = S^{-1}AS$ holds for $S \in \mathbb{C}^{n\times n}$ then A and B are *similar*, which is also denoted by $A \sim B$. If A is similar to a diagonal matrix then it is *diagonalizable*. Let A be Hermitian with the diagonal matrix D having the eigenvalues of A as diagonal entries. Then A is *unitarily diagonalizable*, i.e. $A = UDU^*$ holds with a unitary matrix U and all eigenvalues of A are real. The set $\{x^*Ax \colon x \in \mathbb{C}^n \text{ and } x^*x = 1\}$ is called the *field of values*.

For a matrix $A \in \mathbb{C}^{n\times n}$ a decomposition $U^*AU = T$ with a unitary matrix U and an upper triangular matrix T with the eigenvalues of A on the diagonal is called a *Schur form*. For a real matrix $A \in \mathbb{R}^{n\times n}$ a *real Schur form* is given by $U^*AU = T$ with an orthonormal matrix $U \in \mathbb{R}^{n\times n}$ and a quasi upper triangular matrix $T \in \mathbb{R}^{n\times n}$ with 2×2 blocks on the diagonal which correspond to complex conjugated eigenvalues of A.

Define $\mathcal{J} = \left[\begin{smallmatrix} 0 & I_n \\ -I_n & 0 \end{smallmatrix}\right] \in \mathbb{R}^{2n\times 2n}$. A matrix H is a *Hamiltonian* matrix if $H\mathcal{J}$ is Hermitian. It follows that if $\lambda \in \mathbb{C}$ is an eigenvalue of H then so is $-\overline{\lambda}$ due to the similarity relation $\mathcal{J}^{-1}H\mathcal{J} = -H^*$.

2.2. Projections

Projections [66] are used to reduce the dimension of problems. Let $\Pi \colon \mathbb{C}^n \to \mathbb{C}^n$ be an endomorphism. Then Π is called a *projection* if it is idempotent, i.e. $\Pi^2 = \Pi$. Let

$Z, W \in \mathbb{C}^{n\times j}$ be matrices of rank j, i.e. bases of j-dimensional subspaces of $\mathbb{C}^n$, with regular Z^*W. Then $\Pi = Z(W^*Z)^{-1}W^* \in \mathbb{C}^{n\times n}$ is a projection:

$$\begin{aligned}\Pi^2 &= Z(W^*Z)^{-1}W^*Z(W^*Z)^{-1}W^* \\ &= Z(W^*Z)^{-1}W^*.\end{aligned}$$

We say Π is a projection *onto* $\mathrm{im}(\Pi) = \mathrm{span}(Z)$ *along* $\ker(\Pi) = \ker(W^*) = \mathrm{span}(W)^\perp$. If $\mathrm{im}(\Pi) = \ker(\Pi)^\perp$ holds, then Π is an *orthogonal* projection, otherwise it is an *oblique* projection.

Application of the projection Π to a vector $v \in \mathbb{C}^n$ yields

$$\hat{v} \coloneqq \Pi v = Z(W^*Z)^{-1}W^*v \in \mathrm{span}(Z),$$

which is still an n-dimensional vector. As the projected vector $\hat{v}$ lies in a j-dimensional subspace with basis Z, the dimension can be reduced by using the coordinates determined by this basis. In particular the j-dimensional vector $\tilde{v} = (W^*Z)^{-1}W^*v \in \mathbb{C}^j$ can be used for efficient calculations. The projected n-dimensional vector is regained via $\hat{v} = Z\tilde{v}$.

2.3. Kronecker product

We will frequently make use of the Kronecker product of two matrices as well as the vectorization of a matrix, see, e.g., [44, 35] for a more complete discussion. If X is an $r \times s$ matrix and Y is a $p \times q$ matrix, then the Kronecker product $X \otimes Y$ is the $rp \times sq$ block matrix

$$X \otimes Y = \begin{bmatrix} x_{11}Y & \cdots & x_{1s}Y \\ \vdots & \ddots & \vdots \\ x_{r1}Y & \cdots & x_{rs}Y \end{bmatrix}.$$

For suitable V, W we have $(X \otimes Y)(V \otimes W) = XV \otimes YW$. If X and Y are regular, then the property

$$(X \otimes Y)^{-1} = X^{-1} \otimes Y^{-1}$$

holds.

The vectorization of a matrix converts the matrix into a column vector. For a $r \times s$ matrix X, $\mathrm{vec}(X)$ denotes the $rs \times 1$ column vector obtained by stacking the columns of the matrix X on top of one another:

$$\mathrm{vec}(X) = [x_{11}, \ldots, x_{r1}, x_{12}, \ldots, x_{r2}, \ldots, x_{1s}, \ldots, x_{rs}]^{\mathsf{T}}.$$

The vectorization and the Kronecker product can be used to express matrix multiplication as a linear transformation on matrices. In particular,

$$\mathrm{vec}(XYZ) = (Z^{\mathsf{T}} \otimes X)\,\mathrm{vec}(Y) \tag{2.1}$$

for matrices X, Y, and Z of dimensions $r \times s$, $s \times t$, and $t \times v$. For eigenvalues

$$\lambda(X \otimes Y) = \{\lambda \cdot \mu \mid \lambda \in \lambda(X),\, \mu \in \lambda(Y)\}$$

holds.

2.4. Control theory

Next, concepts from system and control theory are introduced based on [3, 23]. Let $A \in \mathbb{C}^{n \times n}$, $B \in \mathbb{C}^{n \times m}$ and $C \in \mathbb{C}^{p \times n}$. As we only deal with continuous-time problems we define the open left half plane $\mathbb{C}_- = \{z \in \mathbb{C} \colon \mathrm{Re}(z) < 0\}$ as the *stability region*. The matrix A is *stable* if all its eigenvalues lie inside the stability region. If $\mathrm{rank}([A - \lambda I,\, B]) = n$ holds for $\lambda \in \mathbb{C}$ then (A, B) is *controllable at* λ. The tuple (A, B) is *controllable* if $\mathrm{rank}([A - \lambda I,\, B]) = n$ holds for all $\lambda \in \mathbb{C}$. Equivalently (A, B)

is controllable if the *finite reachability matrix* $[B, AB, \ldots, A^{n-1}B]$ has rank n. In the literature also the term *reachable* is used instead of *controllable*. A weaker concept than controllability is *stabilizability*. The tuple (A, B) is called *stabilizable* if it is controllable outside the stability region. There exists a matrix $K \in \mathbb{C}^{m\times n}$ with $A - BK$ stable if and only if (A, B) is stabilizable. The tuple (C, A) is called *observable* if (A^*, C^*) is controllable and *detectable* if (A^*, C^*) is stabilizable. The *finite observability matrix* is given by $[C^*, A^*C^*, \ldots, (A^*)^{n-1}C^*]^*$. If it is of rank n then (C, A) is observable.

2.5. Solution of matrix equations

In this section we discuss the solvability of linear matrix equations and Riccati equations and present algorithms for their solution. The matrix equations considered here can be split into two groups: those with small-scale, dense matrices and those with large-scale, sparse matrices. Further, direct and iterative methods for their solution are distinguished. The main focus of this work is the solution of large-scale problems with iterative methods. However, they rely on the solution of small-scale problems. Therefore direct methods for the solution of small-scale matrix equations are presented here. The following results are based on [23]. For a more detailed discussion on direct and iterative solvers for matrix equations we refer to [63, 44, 50].

2.5.1. Lyapunov and Sylvester equations

The Lyapunov equation

$$A\mathcal{P} + \mathcal{P}A^{\mathsf{T}} + BB^{\mathsf{T}} = 0 \qquad \text{(1.2 revisited)}$$

with $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ is transformed to the linear system of equations

$$(I_n \otimes A + A \otimes I_n)\,\mathrm{vec}(\mathcal{P}) = -\,\mathrm{vec}(BB^{\mathsf{T}}).$$

using (2.1). It follows that the Lyapunov equation has a unique solution if and only if the eigenvalues of A and $-A$ are pairwise distinct. The Sylvester equation

$$A\mathcal{Y} - \mathcal{Y}B - FG^{\mathsf{T}} = 0. \tag{2.2}$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $F \in \mathbb{R}^{n\times r}$ and $G \in \mathbb{R}^{m\times r}$ is rewritten in the same way as

$$(I_m \otimes A - B^{\mathsf{T}} \otimes I_n)\operatorname{vec}(\mathcal{Y}) = \operatorname{vec}(FG^{\mathsf{T}}).$$

A unique solution $\mathcal{Y}$ exists if and only if A and B have no common eigenvalues.

One naive way to obtain the solution of (1.2) or (2.2) is indeed via the solution of these linear systems with a system matrix of dimension n^2 (respectively nm). A more efficient way is to use a variant of the Bartels-Stewart algorithm [4] which we present here for the Lyapunov equation. Let $\tilde{A} = U^*AU$ be a Schur decomposition with an upper triangular matrix $\tilde{A}$. Then (1.2) is equivalent to

$$\tilde{A}\tilde{\mathcal{P}} + \tilde{\mathcal{P}}\tilde{A}^* = -\tilde{B}\tilde{B}^* \tag{2.3}$$

with $\tilde{\mathcal{P}} = U^*\mathcal{P}U$ and $\tilde{B} = U^*B$. The transformed Lyapunov equation (2.3) is solved column by column to obtain $\tilde{\mathcal{P}}$. Therefore set $\tilde{p}_i = \tilde{\mathcal{P}}e_i$ and $\tilde{b}_i = \tilde{B}\tilde{B}^*e_i$. Let

$$\tilde{a}_i = e_i^{\mathsf{T}}\tilde{A} = \begin{bmatrix} 0 & \cdots & 0 & a_{ii} & \cdots & a_{in} \end{bmatrix},$$

so $\tilde{a}_i^*$ is the i-th column of $\tilde{A}^*$. The first $i-1$ entries of $\tilde{a}_i$ are zero due to the triangularity of $\tilde{A}$. For the i-th column of (2.3) we find

$$\begin{aligned} \tilde{A}\tilde{p}_i + \tilde{\mathcal{P}}\tilde{a}_i^* &= -\tilde{b}_i \\ \Leftrightarrow \tilde{A}\tilde{p}_i + \sum_{j=i}^{n} \tilde{p}_j\overline{a_{ij}} &= -\tilde{b}_i \\ \Leftrightarrow \left(\tilde{A} + \overline{a_{ii}}I_n\right)\tilde{p}_i &= -\tilde{b}_i - \sum_{j=i+1}^{n} \tilde{p}_j\overline{a_{ij}}. \end{aligned}$$

These linear systems are solved efficiently for $\tilde{p}_i$, $i = n, \ldots, 1$, via backward substitution as the system matrices are upper triangular. Finally to obtain the solution of the original equation the solution is transformed back via $\mathcal{P} = U\tilde{\mathcal{P}}U^*$. If A is already in triangular form then clearly no Schur decomposition has to be calculated. Modifications of the Bartels-Stewart method presented here exploit the symmetry of the solution $\mathcal{P}$ or use a real Schur form so calculations can be kept real. For Sylvester equations (2.2) the same idea applies with different decompositions for A and B. An even more efficient method for Sylvester equations is the Hessenberg-Schur algorithm [34] where one system matrix is reduced to Hessenberg form instead of Schur form.

2.5.2. Riccati equations

For the Riccati equation

$$A^*X + XA + C^*C - XBB^*X = 0 \qquad \text{(1.1 revisited)}$$

with $A \in \mathbb{C}^{n\times n}$, $B \in \mathbb{C}^{n\times m}$ and $C \in \mathbb{C}^{p\times n}$ many solutions exist due to the quadratic term XBB^*X, see, e.g. [23, Ch. 2]. In most applications one is interested in the unique stabilizing solution. This is the positive semidefinite solution X for which $A - BB^*X$ is stable, i.e. all its eigenvalues are contained in the left half of the complex plane. Existence of this solution is guaranteed if (A, B) is stabilizable and (C, A) is detectable [23, Thm. 2.21], [50, Thm. 9.1.2]. If (C, A) is observable, then the sought solution is even positive definite [50, Thm. 9.1.5].

It is straightforward to see that X is a solution of (1.1) if and only if the invariant subspace equation

$$\begin{bmatrix} A & -BB^* \\ -C^*C & -A^* \end{bmatrix} \begin{bmatrix} I_n \\ X \end{bmatrix} = \begin{bmatrix} I_n \\ X \end{bmatrix} (A - BB^*X)$$

holds with the $2n$-dimensional Hamiltonian matrix

$$H := \begin{bmatrix} A & -BB^* \\ -C^*C & -A^* \end{bmatrix}.$$

Further, if for the n-dimensional matrices $U, V \in \mathbb{C}^{n\times n}$

$$H \begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} U \\ V \end{bmatrix} \Lambda \tag{2.4}$$

holds with $\Lambda \in \mathbb{C}^{n\times n}$, then $X = VU^{-1}$ is a solution of (1.1). Clearly the eigenvalues of Λ are a subset of the eigenvalues of H. Assume (1.1) has a stabilizing solution. As H is Hamiltonian it then has n eigenvalues with negative and n eigenvalues with positive real part. Let (2.4) hold with a stable matrix Λ. Then $X = VU^{-1}$ is the sought-after stabilizing solution. Hence to obtain this solution, an invariant subspace corresponding to the eigenvalues of H with negative real part has to be computed.

To calculate such an invariant subspace let $H = QRQ^*$ be a Schur decomposition with

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}, \quad R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix},$$

where the eigenvalues of R are sorted such that R_{11} is a stable matrix. The stabilizing solution is then given by $X = Q_{21}Q_{11}^{-1}$. In case of a real Riccati equation a real Schur decomposition is used to avoid complex arithmetic.

To increase the accuracy of the above method the matrix H might be preprocessed via similarity transformations to balance the norms of the rows and columns [23, Ch. 3.2.1]. Other methods for the solution of Riccati equations are e.g. Newton's method [23, Ch. 3.3] or the sign function method [23, Ch. 3.5] which calculate the solution iteratively. To refine an approximate solution to the Riccati equation techniques like a few steps of Newton's method or defect correction are available, see [23, Ch. 3.3.4].

2.6. Numerical examples

In the numerical experiments we will make use of the following benchmark examples. They come from real applications and are widely used for the comparison of numerical algorithms. All examples are available from the UF Sparse Matrix Collection [28] via the given ID.

The rail example (ID 1445, [15]) describes the semi-discretization of a heat transfer process for optimal cooling of steel profiles. Due to different mesh widths used in the semi-discretization matrices of several dimensions are available. We chose the largest example which consists of symmetric negative/positive definite $A, E \in \mathbb{R}^{79841\times 79841}$, $B \in \mathbb{R}^{79841\times 7}$ and $C \in \mathbb{R}^{6\times 79841}$. It is denoted by *rail79k*.

The second example is the *chip0* example (ID 1428, [55]), a finite element model of a chip cooled by convection. The system matrices are $A, E \in \mathbb{R}^{20082\times 20082}$, $B \in \mathbb{R}^{20082\times 1}$ and $C \in \mathbb{R}^{5\times 20082}$. Both originate from the Oberwolfach Benchmark Collection [47].

Further, we use the example *lung2* (ID 894), modeling processes in the human lung. We employ this example with the negated system matrix $-A \in \mathbb{R}^{109460\times 109460}$, $E = I$ and $C^* = B \in \mathbb{R}^{109460\times 10}$ chosen at random.

3. Rational Krylov subspaces and rational Arnoldi decompositions

We proceed with a discussion on (rational) Krylov subspaces which are the used approximation spaces in the following parts. The definitions and theorems in this section are based on [18, 16]. Given a matrix $A \in \mathbb{C}^{n\times n}$, a vector $b \in \mathbb{C}^n$ and a number $j \in \mathbb{N}$, the *polynomial Krylov subspace of order* j is defined by

$$\begin{aligned} \mathcal{K}_j(A, b) &:= \operatorname{span}\Big(b, Ab, \ldots, A^{j-1}b\Big) \\ &= \{\mathfrak{p}(A)b \mid \mathfrak{p} \in \mathbf{\Pi}_{(j-1)}\}, \end{aligned} \tag{3.1}$$

i.e. the set of all polynomials of degree at most $j-1$ in A multiplied with b. We extend this definition to rational functions instead of only polynomials. Let $\mathfrak{q} \in \mathbf{\Pi}_{(j)}$ be a fixed polynomial of degree at most j with no roots in $\lambda(A)$ so $\mathfrak{q}(A)$ is invertible. The *rational Krylov subspace of order j associated to* $\mathfrak{q}$ is defined by

$$\begin{aligned} \mathcal{K}_j(A, b, \mathfrak{q}) &:= \mathfrak{q}(A)^{-1}\mathcal{K}_j(A, b) \\ &= \{\mathfrak{q}(A)^{-1}\mathfrak{p}(A)b \mid \mathfrak{p} \in \mathbf{\Pi}_{(j-1)}\}, \end{aligned} \tag{3.2}$$

i.e. the set of all rational functions in A multiplied with b with a numerator polynomial of degree at most $j-1$ and fixed denominator polynomial $\mathfrak{q}$ of degree equal or less than the order j of the Krylov subspace $\mathcal{K}_j(A, b)$. The polynomial $\mathfrak{q}$ has roots $s_1, \ldots, s_{\deg(\mathfrak{q})} \in \mathbb{C} \setminus \lambda(A)$. If $\deg(\mathfrak{q}) < j$ we say it has $j - \deg(\mathfrak{q})$ formal roots at infinity. This allows us to characterize the rational Krylov subspace through a set of j roots $s = \{s_1, \ldots, s_j\} \subset \overline{\mathbb{C}} \setminus \lambda(A)$ instead of the polynomial $\mathfrak{q}$, i.e. we can write $\mathcal{K}_j(A, b, s)$ for (3.2). The roots of $\mathfrak{q}$ are also called *shifts* or *poles* of the rational Krylov subspace.

The polynomial Krylov subspace (3.1) of order j is a special case of a rational Krylov subspace with $\mathfrak{q} = 1$, i.e. a rational Krylov subspace of order j with j poles at infinity. On the other extreme, if $\mathfrak{q}$ has degree j, then $\mathfrak{q}^{-1}\mathfrak{p}$ is a proper rational function as the degree of $\mathfrak{p}$ is at most $j-1$ and so $b \notin \mathcal{K}_j(A, b, \mathfrak{q})$ holds. In between these two cases we may apply polynomial long division to decompose the rational Krylov subspace. Let $\mathfrak{p} \in \mathbf{\Pi}_{(j-1)}$, $\mathfrak{q} \in \mathbf{\Pi}_{(j)}$, and set $l = j - \deg(\mathfrak{q})$, the number of poles at infinity of a rational Krylov subspace of order j associated to $\mathfrak{q}$. We find $\mathfrak{u} \in \mathbf{\Pi}_{(l-1)}$ and $\mathfrak{r} \in \mathbf{\Pi}_{(\deg(\mathfrak{q})-1)}$ with $\mathfrak{p} = \mathfrak{q}\mathfrak{u} + \mathfrak{r}$ which results in the decomposition

$$\begin{aligned} \mathcal{K}_j(A, b, \mathfrak{q}) &= \{\mathfrak{u}(A)b + \mathfrak{q}(A)^{-1}\mathfrak{r}(A)b \mid \mathfrak{u} \in \mathbf{\Pi}_{(l-1)},\, \mathfrak{r} \in \mathbf{\Pi}_{(\deg(\mathfrak{q})-1)}\} \\ &= \mathcal{K}_l(A, b) + \mathcal{K}_{\deg(\mathfrak{q})}(A, b, \mathfrak{q}) \end{aligned} \tag{3.3}$$

of a rational Krylov subspace into a polynomial Krylov subspace of order l and a rational Krylov subspace of order $\deg(\mathfrak{q})$ without poles at infinity.

In particular it holds $\mathcal{K}_l(A, b) \subset \mathcal{K}_j(A, b, s)$ if s contains the shift infinity at least l

times. More specifically b is contained in a rational Krylov subspace if the shift infinity occurs at least once. We often omit the term *rational* in the following.

Consider the general case of an A-variant Krylov subspace $\mathcal{K}_j(A, b, s)$, that is,

$$A\mathcal{K}_j(A, b, s) \not\subseteq \mathcal{K}_j(A, b, s).$$

Then the spaces $\mathcal{K}_j(A, b, s)$ and $A\mathcal{K}_j(A, b, s)$ are of dimension j and their intersection is of dimension $j-1$, i.e. $\mathcal{K}_j(A, b, s)$ is almost A-invariant. This implies that the sum of these two spaces is of dimension $j+1$. More specifically it holds

$$\begin{aligned} A\mathcal{K}_j(A, b, s) &= A\mathcal{K}_j(A, b, \mathfrak{q}) \\ &= \mathfrak{q}(A)^{-1} A\mathcal{K}_j(A, b) \\ &\subset \mathfrak{q}(A)^{-1}\mathcal{K}_{j+1}(A, b) \\ &= \mathcal{K}_{j+1}(A, b, \mathfrak{q}) \\ &= \mathcal{K}_{j+1}(A, b, s \cup \{\infty\}), \end{aligned} \tag{3.4}$$

with one additional shift infinity in the last line because the order of the polynomial Krylov subspace was increased by one without altering $\mathfrak{q}$. We call the latter space with an additional infinite shift *augmented Krylov subspace* and use the notation $\mathcal{K}_j^+(A, b, s) \coloneqq \mathcal{K}_{j+1}(A, b, s \cup \{\infty\})$. It contains rational functions in A multiplied with b whose numerator polynomials are of degree at most j instead of only $j-1$ as in (3.2).

Let V_{j+1} be a basis of the augmented space $\mathcal{K}_j^+(A, b, s)$. As it contains both spaces $\mathcal{K}_j(A, b, s)$ and $A\mathcal{K}_j(A, b, s)$ it is possible to encode the effect of multiplication of $\mathcal{K}_j(A, b, s)$ with A in a decomposition of the form

$$AV_{j+1}\underline{K_j} = V_{j+1}\underline{H_j} \tag{3.5}$$

with rectangular matrices $\underline{K_j}$, $\underline{H_j} \in \mathbb{C}^{(j+1)\times j}$ and

$$\begin{aligned} \operatorname{span}(V_{j+1}\underline{K_j}) &= \mathcal{K}_j(A, b, s), \\ \operatorname{span}(V_{j+1}\underline{H_j}) &= A\mathcal{K}_j(A, b, s). \end{aligned} \tag{3.6}$$

Decompositions of the form (3.5), their connections to Krylov subspaces and their properties play a key role in our discussion on Riccati equations. We therefore introduce some central definitions and results from [18].

Remark 3.1. We note that our definition (3.2) of a rational Krylov subspace differs from the definition in [18] in the order of the involved polynomial Krylov subspace. The rational Krylov subspaces defined in [18] are equivalent to our augmented Krylov subspace (3.4). The reason for the deviation from the literature is that for our purposes Krylov subspaces with only finite shifts are needed, but the augmented Krylov subspaces always contain (at least) one infinite shift. See also [16, Prop. 3.3] and the paragraph preceding it.

Definition 3.2 (cf. [18, Def. 4.1])**.** Let $\underline{K_j}, \underline{H_j} \in \mathbb{C}^{(j+1)\times j}$ be rectangular matrices. We say that the pencil $(\underline{H_j}, \underline{K_j})$ is *regular* if the lower $j \times j$ subpencil $(\underline{H_{-j}}, \underline{K_{-j}})$ is regular, i.e., $\det(z\underline{K_{-j}} - \underline{H_{-j}})$ is not identically equal to zero.

Definition 3.3 ([18, Def. 4.2])**.** A relation of the form (3.5) where V_{j+1} is of full column rank and $(\underline{H_{-j}}, \underline{K_{-j}})$ is regular is called a *generalized rational Krylov decomposition*. The generalized eigenvalues of $(\underline{H_{-j}}, \underline{K_{-j}})$ are called *poles of the decomposition*. If the poles of (3.5) are outside the spectrum $\lambda(A)$, then (3.5) is called a *rational Krylov decomposition (RKD)*.

If in a (generalized) RKD the matrices $\underline{K_j}$ and $\underline{H_j}$ are upper-Hessenberg matrices then the decomposition is called a *(generalized) rational Arnoldi decomposition (RAD)*. The columns of V_{j+1} are called the *basis of the decomposition* and they span the *augmented space of the decomposition* (cf. [18, Def. 2.3]). The first column of V_{j+1} is called *starting vector*. Every (generalized) RKD can be transformed to a (generalized) RAD with the same starting vector and the same poles using a generalized Schur form of $(\underline{H_{-j}}, \underline{K_{-j}})$

(see [18, Thm. 4.3]). If A and b are real valued and the poles appear in complex conjugated pairs then a generalized real Schur form of $(\underline{H_{-j}}, \underline{K_{-j}})$ can be used to obtain a *quasi-RAD*, that is $\underline{K_{-j}}$ is a real upper triangular matrix and $\underline{H_{-j}}$ is a real quasi upper triangular matrix with 1-by-1 and 2-by-2 blocks on the diagonal (cf. [16, Def. 2.17]). The following theorem guarantees the existence of a RAD for a Krylov subspace (3.2).

Theorem 3.4 (cf. [18, Thm. 2.5])**.** *Let $\mathcal{V}_{j+1}$ be a vector space of dimension $j+1$ and $s \subset \overline{\mathbb{C}}$ be a set with j elements. Then $\mathcal{V}_{j+1} = \mathcal{K}_j^+(A, b, s)$ holds if and only if there exists a RAD $AV_{j+1}\underline{K_j} = V_{j+1}\underline{H_j}$ with $\underline{K_j}$, $\underline{H_j} \in \mathbb{C}^{(j+1)\times j}$, starting vector $b = V_{j+1}e_1$, poles s and* $\operatorname{span}(V_{j+1}) = \mathcal{V}_{j+1}$.

Let $\mathcal{K}_j(A, b, s)$ be a Krylov subspace with associated RKD

$$AV_{j+1}\underline{K_j} = V_{j+1}\underline{H_j}. \tag{3.7}$$

Such a RKD can be transformed by a regular matrix $U \in \mathbb{C}^{(j+1)\times(j+1)}$ into the (generalized) RKD

$$A\check{V}_{j+1}\underline{\check{K}_j} = \check{V}_{j+1}\underline{\check{H}_j} \tag{3.8}$$

with $\check{V}_{j+1} = V_{j+1}U$, $\underline{\check{K}_j} = U^{-1}\underline{K_j}$ and $\underline{\check{H}_j} = U^{-1}\underline{H_j}$. The RKD (3.8) is associated to the same Krylov subspace $\mathcal{K}_j(A, b, s)$ as (3.7). In general the starting vector $\check{b} = \check{V}_{j+1}e_1 = V_{j+1}Ue_1$ and the poles $\lambda(\underline{\check{H}_{-j}}, \underline{\check{K}_{-j}})$ are altered through this transformation and it may happen that the poles coincide with eigenvalues of A. Moreover, as $\check{b} \in \mathcal{K}_j^+(A, b, s)$, there is a polynomial $\mathfrak{q} \in \mathbf{\Pi}_{(j)}$ with roots s and a polynomial $\check{\mathfrak{q}} \in \mathbf{\Pi}_{(j)}$ such that $\check{b} = \mathfrak{q}(A)^{-1}\check{\mathfrak{q}}(A)b$. Thus, if no root of $\check{\mathfrak{q}}$ coincides with an eigenvalue of A then

$\check{\mathfrak{q}}(A)^{-1}\mathfrak{q}(A)\check{b} = b$ holds. This implies

$$\begin{aligned}\mathcal{K}_j(A,b,\mathfrak{q}) &= \mathfrak{q}(A)^{-1}\mathcal{K}_j(A,b)\\ &= \{\mathfrak{q}(A)^{-1}\mathfrak{p}(A)b \mid \mathfrak{p}\in\mathbf{\Pi}_{(j-1)}\}\\ &= \{\mathfrak{q}(A)^{-1}\mathfrak{p}(A)\check{\mathfrak{q}}(A)^{-1}\mathfrak{q}(A)\check{b} \mid \mathfrak{p}\in\mathbf{\Pi}_{(j-1)}\}\\ &= \{\check{\mathfrak{q}}(A)^{-1}\mathfrak{p}(A)\check{b} \mid \mathfrak{p}\in\mathbf{\Pi}_{(j-1)}\}\\ &= \mathcal{K}_j(A,\check{b},\check{\mathfrak{q}}),\end{aligned}$$

where we have used the commutativity of rational functions in A. Even if a root of $\check{\mathfrak{q}}$ coincides with an eigenvalue of A the following result holds.

Theorem 3.5 (cf. [18, Thm. 4.4])**.** *Let $\mathcal{V}_{j+1} = \mathcal{K}_j^+(A,b,\mathfrak{q})$ be A-variant. Let $\check{\mathfrak{q}}\in\mathbf{\Pi}_{(j)}$ be a polynomial with roots equal to the poles of the (generalized) RKD $A\check{V}_{j+1}\underline{\check{K}_j} = \check{V}_{j+1}\underline{\check{H}_j}$. If $\check{V}_{j+1}$ spans $\mathcal{V}_{j+1}$, then for the starting vector $\check{b} = \check{V}_{j+1}e_1$ it holds $\check{b} = \gamma\mathfrak{q}(A)^{-1}\check{\mathfrak{q}}(A)b$ with a scalar $0\neq\gamma\in\mathbb{C}$.*

Let $0\neq\check{b}\in\mathcal{K}_j^+(A,b,\mathfrak{q})$ be an arbitrary element of the augmented Krylov subspace with associated RKD (3.7). Thus, there is a polynomial $\check{\mathfrak{q}}\in\mathbf{\Pi}_{(j)}$ with $\check{b} = \mathfrak{q}(A)^{-1}\check{\mathfrak{q}}(A)b$ and a nontrivial vector u_1 with $\check{b} = V_{j+1}u_1$. Let U be a regular matrix with $Ue_1 = u_1$, so $\check{b} = V_{j+1}Ue_1$ holds. Herewith Theorem 3.5 allows us to determine the roots of the numerator polynomial $\check{\mathfrak{q}}$ as the poles of the RKD (3.8), i.e. the generalized eigenvalues of $(\underline{\check{H}_{-j}},\underline{\check{K}_{-j}})$.

The definition of rational Krylov subspaces and rational Arnoldi decompositions can be generalized to the case where the vector $b\in\mathbb{C}^n$ is replaced by a matrix (or *block vector*) $\boldsymbol{b}\in\mathbb{C}^{n\times p}$. We recall some of the important definitions based on [30]. The block Krylov subspace of order j is given by

$$\begin{aligned}\mathcal{K}_j^{\square}(A,\boldsymbol{b}) &= \operatorname{blockspan}\left\{\boldsymbol{b}, A\boldsymbol{b},\ldots,A^{j-1}\boldsymbol{b}\right\}\\ &= \left\{\sum_{k=0}^{j-1}A^k\boldsymbol{b}C_k \mid C_k\in\mathbb{C}^{p\times p}\right\}.\end{aligned}\tag{3.9}$$

We only consider the case where the jp columns of $[\boldsymbol{b}, A\boldsymbol{b}, \ldots, A^{j-1}\boldsymbol{b}]$ are linearly independent so the block Krylov subspace (3.9) has dimension jp^2. Every block vector $\sum_{k=0}^{j-1} A^k \boldsymbol{b} C_k \in \mathcal{K}_j^{\square}(A, \boldsymbol{b})$ corresponds to exactly one matrix polynomial $\sum_{k=0}^{j-1} z^k C_k$.

For the definition of a *block rational Krylov subspace* we again use a polynomial $\mathfrak{q} \in \boldsymbol{\Pi}_{(j)}$ of degree at most j with no roots in $\lambda(A)$ and set

$$\mathcal{K}_j^{\square}(A, \boldsymbol{b}, \mathfrak{q}) = \mathfrak{q}(A)^{-1} \mathcal{K}_j^{\square}(A, \boldsymbol{b}).$$

As in the non-block part we set $\mathcal{K}_j^{\square}(A, \boldsymbol{b}, s) = \mathcal{K}_j^{\square}(A, \boldsymbol{b}, \mathfrak{q})$ with the set $s \subset \overline{\mathbb{C}}$ of the j roots of $\mathfrak{q}$.

Before we can define block rational Arnoldi decompositions we need the following definition of a block upper-Hessenberg matrix.

Definition 3.6 (cf. [30, Def. 2.1])**.** The block matrix

$$\underline{H_j} = \begin{bmatrix} H_{11} & \cdots & H_{1p} \\ H_{21} & \cdots & H_{2p} \\ & \ddots & \vdots \\ & & H_{j+1,j} \end{bmatrix} \in \mathbb{C}^{(j+1)p \times jp}, \ H_{ik} \in \mathbb{C}^{p \times p}$$

is called a *block upper-Hessenberg matrix*. For block upper-Hessenberg matrices $\underline{H_j}$ and $\underline{K_j}$ the pencil $(\underline{H_j}, \underline{K_j})$ is called an *unreduced block upper-Hessenberg pencil* if one of the subdiagonal blocks $H_{i+1,i}$ or $K_{i+1,i}$ is nonsingular for every $i = 1, \ldots, j$.

The next definition generalizes RADs to the block case.

Definition 3.7 (see [30, Def. 2.2])**.** Let $A \in \mathbb{C}^{n \times n}$. A relation of the form

$$AV_{j+1}\underline{K_j} = V_{j+1}\underline{H_j}$$

is called a *block rational Arnoldi decomposition* (BRAD) if the following conditions are satisfied:

1. V_{j+1} is of full column rank,

2. $(\underline{H_j}, \underline{K_j})$ is an unreduced block upper-Hessenberg pencil,

3. $\alpha_i K_{i+1,i} = \beta_i H_{i+1,i}$ holds for some scalars $\alpha_i, \beta_i \in \mathbb{C}$ not both zero,

4. the numbers $\mu_i = \alpha_i / \beta_i$ are outside the spectrum $\lambda(A)$

for $i = 1, \ldots, j$. The numbers $\mu_i \in \overline{\mathbb{C}}$ are called the *poles* of the BRAD.

One important class of (block) rational Arnoldi decompositions are the *orthonormal (B)RADs* where the basis V_{j+1} has orthonormal columns. To calculate such an orthonormal decomposition the rational Krylov method by Ruhe can be used. It is sketched here for orthonormal RADs in Algorithm 3.1 based on the formulation used in [16]. First the starting vector is normalized. Then the orthonormal basis is created iteratively. The space spanned by $V_i \in \mathbb{C}^{n \times i}$ is extended by the vector w_{i+1} which is obtained in line 4. The tuple $(\eta_i / \rho_i, t_i) \in \overline{\mathbb{C}} \times \mathbb{C}^i$ has to be an *admissible continuation pair* which means that $w_{i+1} \notin \operatorname{span}(V_i)$ holds and the space is indeed enlarged. In lines 5 and 6 orthonormalization takes place. In line 7 the last columns of $\underline{K_i}$ and $\underline{H_i}$ are assembled. The notation $s_i = \mu_i / \nu_i$ allows to handle infinite poles by using $\mu_i \neq 0$ and $\nu_i = 0$. For real A and b there exist variants of Algorithm 3.1 which keep the basis V_{j+1} real in case of poles occurring in complex conjugated pairs, see [16, Ch. 2.4]. Another variant of the algorithm which makes it possible to parallelize the solution of the linear systems is presented in [19].

Algorithm 3.1 Rational Arnoldi algorithm (cf. [16, Alg. 2.2])

Input: $A \in C^{n \times n}$, $b \in \mathbb{C}^{n \times 1}$, poles $s \subset \overline{\mathbb{C}} \setminus \lambda(A)$ with $s_i = \mu_i / \nu_i$
Output: decomposition $A V_{j+1} \underline{K_j} = V_{j+1} \underline{H_j}$ with V_{j+1} orthonormal

1: normalize $v_1 = b / \|b\|_2$
2: **for** $i = 1, \ldots, j$ **do**
3: choose an admissible continuation pair $(\eta_i / \rho_i, t_i) \in \overline{\mathbb{C}} \times \mathbb{C}^i$
4: compute $w_{i+1} = (\nu_i A - \mu_i I_n)^{-1} (\rho_i A - \eta_i I_n) V_i t_i$
5: orthogonalize $\hat{v}_{i+1} = w_{i+1} - V_i c_i$, where $c_i = V_i^* w_{i+1}$
6: normalize $v_{i+1} = \hat{v}_{i+1} / c_{i+1,i}$, where $c_{i+1,i} = \|\hat{v}_{i+1}\|_2$
7: set $\underline{k_i} = \nu_i \underline{c_i} - \rho_i \underline{t_i}$ and $\underline{h_i} = \mu_i \underline{c_i} - \eta_i \underline{t_i}$, where $\underline{t_i} = \begin{bmatrix} t_i \\ 0 \end{bmatrix}$ and $\underline{c_i} = \begin{bmatrix} c_i \\ c_{i+1,i} \end{bmatrix}$
8: **end for**

4. Evolution of the ADI method

The Alternating direction implicit (ADI) method is an iterative method suited for the solution of large-scale matrix equations with sparse system matrices. We briefly discuss some important steps in its evolution mainly based on [23, 48, 12].

The ADI iteration was developed as a method for the numerical solution of partial differential equations in [58]. Computation of the solution for the Lyapunov equation

$$A\mathcal{P} + \mathcal{P}A^{\mathsf{T}} + BB^{\mathsf{T}} = 0 \qquad \text{(1.2 revisited)}$$

is done via the iteration

$$\begin{aligned} X_0 &= 0, \\ (A + \alpha_j I_n) X_{j-\frac{1}{2}} &= -BB^{\mathsf{T}} - X_{j-1}(A^{\mathsf{T}} - \alpha_j I_n), \\ (A + \alpha_j I_n) X_j^* &= -BB^{\mathsf{T}} - X_{j-\frac{1}{2}}^*(A^{\mathsf{T}} - \alpha_j I_n), \end{aligned}$$

with complex shift parameters $\alpha_1, \ldots, \alpha_N \in \mathbb{C}_-$. For a certain choice of shift parameters the iterates X_j will converge to $\mathcal{P}$. In every step a linear system with $2n$ columns on the right-hand side has to be solved.

To make the iteration more efficient the low rank factorization BB^{T} is exploited as in [59]. The iterates are rewritten as $X_j = Z_j Z_j^*$ which is possible as the X_j are positive semidefinite. The resulting iteration reads

$$\begin{aligned} Z_1 &= \sqrt{-2\operatorname{Re}(\alpha_1)}(A + \alpha_1 I_n)^{-1} B, \\ Z_j &= (A + \alpha_j I_n)^{-1} \left[\sqrt{-2\operatorname{Re}(\alpha_j)} B \;\; (A - \alpha_j I_n) Z_{j-1} \right]. \end{aligned}$$

The approximate Cholesky factors Z_j grow by m columns in every step, so a linear system with jm columns on the right-hand side has to be solved in step j.

It was found in [51] that the iteration can be reformulated so only linear systems with m columns have to be solved in each step. The iteration in the formulation of [48,

Algorithm 4.1 Low rank ADI iteration [48, Alg. 3.2, $E = I_n$]

Input: $A \in \mathbb{R}^{n\times n}$ stable, $B \in \mathbb{R}^{n\times m}$, parameters $\{\alpha_1, \dots, \alpha_N\} \in \mathbb{C}_-$
Output: $Z \in \mathbb{C}^{n\times mN}$ with $ZZ^* \approx \mathcal{P}$

1: initialize $W_0 = B$, $Z_0 = [\,]$
2: **for** $j = 1, \dots, N$ **do**
3: solve $(A + \alpha_j I_n)V_j = W_{j-1}$ for V_j
4: $W_j = W_{j-1} - 2\operatorname{Re}(\alpha_j)V_j$
5: update $Z_j = [Z_{j-1}, \sqrt{-2\operatorname{Re}(\alpha_j)}V_j]$
6: **end for**
7: $Z = Z_N$

Ch. 3.2] is given by

$$
\begin{aligned}
V_1 &= (A + \alpha_1 I_n)^{-1} B,\ Z_1 = \sqrt{-2\operatorname{Re}(\alpha_1)}V_1,\\
V_j &= V_{j-1} - (\alpha_j + \overline{\alpha_{j-1}})(A + \alpha_j I_n)^{-1} V_{j-1},\\
Z_j &= \left[Z_{j-1}, \sqrt{-2\operatorname{Re}(\alpha_j)}V_j\right].
\end{aligned}
$$

It was shown in [51] that the columns of Z_j span a (block) rational Krylov subspace with poles $-\alpha_i$, $i = 1, \dots, j$, that is $\operatorname{span}(Z_j) = \mathcal{K}_j^{\square}(A, B, \{-\alpha_1, \cdots, -\alpha_j\})$. See also [5, Sec. 2.4] for a different proof.

In [48, Ch. 3.2.4] the iteration was refined and resulted in the residual based formulation which is stated in Algorithm 4.1. It holds $\mathcal{L}(Z_j Z_j^*) = W_j W_j^*$ and so the residual is of rank at most m. The norm of the residual can be evaluated efficiently via $\|W_j W_j^*\| = \|W_j^* W_j\|$, where the latter matrix is of dimension m. Note that the residual based iteration was independently found in [69].

As the shift parameters α_j are complex numbers, the iterates V_j (and thus Z_j) are complex-valued matrices. This can be avoided if the set of shift parameters is proper, i.e. closed under complex conjugation such that complex shift parameters appear in pairs with their complex conjugated version. We briefly present the realification approach from [13] in the revised form of [48, Ch. 4.1.4]. It is denoted M4 in [48, Ch. 4.1.5]. Let α_j be a complex shift and $\alpha_{j+1} = \overline{\alpha_j}$. Instead of performing two separate steps, one with the shift α_j and one with α_{j+1}, a double step involving α_j and α_{j+1} will be used. That

is, the block $\hat{Z} = \sqrt{-2\operatorname{Re}(\alpha_j)}[V_j, V_{j+1}]$ of $2m$ columns is added to the current iterate Z_{j-1}. This is still a complex matrix, but it can be replaced by a real one. To this end, note that

$$V_{j+1} = \overline{V}_j + 2\frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)} \operatorname{Im}(V_j).$$

Let

$$J = \begin{bmatrix} 1 & 1 \\ \imath & 2\frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)} - \imath \end{bmatrix} \text{ and } L = \sqrt{2} \begin{bmatrix} 1 & 0 \\ \frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)} & \sqrt{\frac{\operatorname{Re}(\alpha_j)^2}{\operatorname{Im}(\alpha_j)^2} + 1} \end{bmatrix}. \tag{4.1}$$

Then the block $\hat{Z}$ of $2m$ columns that is added to the factor Z_{j-1} can be written as

$$\hat{Z} = \sqrt{-2\operatorname{Re}(\alpha_j)} \begin{bmatrix} \operatorname{Re}(V_j) & \operatorname{Im}(V_j) \end{bmatrix} (J \otimes I_m).$$

Observing that

$$JJ^* = \begin{bmatrix} 2 & 2\frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)} \\ 2\frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)} & 4\frac{\operatorname{Re}(\alpha_j)^2}{\operatorname{Im}(\alpha_j)^2} + 2 \end{bmatrix} = LL^*,$$

we can use $\check{Z} = \sqrt{-2\operatorname{Re}(\alpha_j)}[\operatorname{Re}(V_j), \operatorname{Im}(V_j)] (L \otimes I_m)$ instead of $\hat{Z}$, as $\check{Z}\check{Z}^* = \hat{Z}\hat{Z}^*$.

Altogether, to keep the iterates real, the columns

$$\sqrt{-2\operatorname{Re}(\alpha_j)} \cdot \sqrt{2} \begin{bmatrix} \operatorname{Re}(V_j) + \frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)} \operatorname{Im}(V_j) & \sqrt{\frac{\operatorname{Re}(\alpha_j)^2}{\operatorname{Im}(\alpha_j)^2} + 1} \cdot \operatorname{Im}(V_j) \end{bmatrix}$$

are added to the iterate Z_{j-1}, yielding the same approximation Z_{j+1} as two steps of Algorithm 4.1 with shifts α_j and $\alpha_{j+1} = \overline{\alpha_j}$, but with real approximate Cholesky factors.

For a discussion of different strategies for choosing the parameters α_j the reader is referred to [12]. We refrain from stating the ADI iteration for solving the Sylvester equation (2.2) as we will not make explicit use of it. Consult e.g. [48] for a detailed description of a residual based variant. For the ADI iteration for Sylvester equations we refer to [14], [48, Ch. 3.3] and [63]. An introduction to ADI-type iterations for Riccati equations is given in Section 7.

5. Model order reduction

In model order reduction one is interested in replacing large-scale dynamical systems with smaller ones which show a similar behavior. The problem addressed here is the approximation of the stable, linear time invariant (LTI) continuous time dynamical system

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) \end{aligned} \tag{5.1}$$

by the smaller system

$$\begin{aligned} \dot{\hat{x}}(t) &= \hat{A}\hat{x}(t) + \hat{B}u(t), \\ \hat{y}(t) &= \hat{C}\hat{x}(t) \end{aligned} \tag{5.2}$$

with possibly complex reduced system matrices $\hat{A} \in \mathbb{C}^{r\times r}$, $\hat{B} \in \mathbb{C}^{r\times 1}$, $\hat{C} \in \mathbb{C}^{1\times r}$ and $r \ll n$. The reduced dimension of the system matrices makes computations with this system faster than with the original system. As we assume that the system matrices in (5.1) are known we will pursue a projection-based approach: two $n \times r$ matrices $V, W \in \mathbb{C}^{n\times r}$ with $W^*V = I_r$ are computed which define the projector $\Pi = VW^*$. The projection of the states of the original system generates the reduced-order model with system matrices

$$\hat{A} = W^*AV, \qquad \hat{B} = W^*B, \qquad \hat{C} = CV.$$

In practice it is often desirable to have a real reduced system which can be obtained by only using real projections. For ease of notation and explanation here we allow for complex valued projections. For an overview of different model order reduction techniques we refer the reader to [10] and the references therein.

5.1. Balanced truncation

The concept of balancing is related to the energy necessary to reach and observe a state. To measure this energy the controllability Gramian $\mathcal{P}$ and the observability Gramian $\mathcal{Q}$ of the system (5.1) are utilized [3, 54, 49]. The Gramians are defined as

$$\begin{aligned} \mathcal{P} &= \int_0^\infty e^{At} B B^\mathsf{T} e^{A^\mathsf{T} t} \,\mathrm{d}t \in \mathbb{R}^{n\times n}, \\ \mathcal{Q} &= \int_0^\infty e^{A^\mathsf{T} t} C^\mathsf{T} C e^{At} \,\mathrm{d}t \in \mathbb{R}^{n\times n}. \end{aligned} \tag{5.3}$$

In a balanced LTI system states are equally well reachable and observable. To obtain a balanced system observe that the Gramians $\mathcal{P}$ and $\mathcal{Q}$ are positive definite matrices as all eigenvalues of A have negative real part. Thus, their Cholesky decompositions $\mathcal{P} = SS^\mathsf{T}$ and $\mathcal{Q} = RR^\mathsf{T}$ can be determined. Let $U\Sigma T^\mathsf{T}$ be a singular value decomposition of $R^\mathsf{T}S$. Then $F = \Sigma^{-\frac{1}{2}} U^\mathsf{T} R^\mathsf{T}$ and $F^{-1} = ST^{-\mathsf{T}}\Sigma^{-\frac{1}{2}}$ define a balancing transformation. That is, the Gramians $\hat{\mathcal{P}} = F\mathcal{P}F^\mathsf{T}$ and $\hat{\mathcal{Q}} = F^{-\mathsf{T}}\mathcal{Q}F^{-1}$ of the transformed system

$$\begin{aligned} \dot{x}(t) &= FAF^{-1}x(t) + FBu(t), \quad x(0) = x_0, \\ y(t) &= CF^{-1}x(t), \end{aligned}$$

are equal and diagonal. Thus, in the balanced system it holds $\hat{\mathcal{P}} = \hat{\mathcal{Q}} = \Sigma$. The square roots of the eigenvalues of $\mathcal{P}\mathcal{Q}$ are the *Hankel singular values*. They are used as an indicator for the importance of the corresponding state and invariant under state space transformations, i.e. the same for $\mathcal{P}\mathcal{Q}$ and $\hat{\mathcal{P}}\hat{\mathcal{Q}} = F\mathcal{P}\mathcal{Q}F^{-1}$ due to similarity. In the balanced system the Hankel singular values are the diagonal entries of Σ.

To reduce the order of the balanced system we use balanced truncation. A projection is performed onto the r most important states, i.e. the states with large Hankel singular values. The projection $\Pi = VW^\mathsf{T}$ for the reduction process is derived from the partitioned singular value decomposition

$$R^\mathsf{T}S = \begin{bmatrix} U_r & U_0 \end{bmatrix} \begin{bmatrix} \Sigma_r & \\ & \Sigma_0 \end{bmatrix} \begin{bmatrix} T_r^\mathsf{T} \\ T_0^\mathsf{T} \end{bmatrix}$$

with $\Sigma_r \in \mathbb{R}^{r\times r}$, $U_r \in \mathbb{R}^{n\times r}$ and $T_r \in \mathbb{R}^{n\times r}$. The matrices V and W are obtained as

$$W = RU_r\Sigma_r^{-\frac{1}{2}},\ V = ST_r\Sigma_r^{-\frac{1}{2}} \tag{5.4}$$

and indeed $W^\mathsf{T}V = I_r$ holds. For more details on the Gramians and the energy associated with reaching/observing a state see, e.g., [3, Ch. 4.3].

A bottleneck in this approach is the calculation of the Gramians $\mathcal{P}$, $\mathcal{Q}$ and their Cholesky factors. To make calculations for large systems computationally feasible the Gramians are approximated with low rank factors, i.e. $Z_\mathrm{c}Z_\mathrm{c}^\mathsf{T} \approx \mathcal{P}$ and $Z_\mathrm{o}Z_\mathrm{o}^\mathsf{T} \approx \mathcal{Q}$ with rectangular matrices $Z_\mathrm{c} \in \mathbb{R}^{n\times r_\mathrm{c}}$, $Z_\mathrm{o} \in \mathbb{R}^{n\times r_\mathrm{o}}$, and $r_\mathrm{c}, r_\mathrm{o} \ll n$. These approximate Cholesky factors Z_c and Z_o are then used instead of the actual Cholesky factors S and R to compute an approximate balancing transformation. This also includes a reduction of the system dimension as the number of columns in the approximate Cholesky factors is smaller than n. In balanced truncation with the actual Cholesky factors S and R of the Gramians as described above, stability of the system is preserved and there exists an error bound in terms of the truncated Hankel singular values [3, Thm. 7.9]. When the approximate Cholesky factors Z_c and Z_o are used, these properties are lost. However, in practice often the reduced models are stable even when approximate Cholesky factors are used, see e.g. [38, Sec. 4.3].

5.2. Rational interpolation

In rational interpolation, also called moment matching, [3, 31, 37] the projection matrices V and W are chosen so that the transfer function $G(s) = C(sI_n - A)^{-1}B$ of the original system (5.1) and the transfer function $\hat{G}(s) = \hat{C}(sI_r - \hat{A})^{-1}\hat{B}$ of the reduced system (5.2) (and some of their derivatives) coincide at certain interpolation points $s \in \overline{\mathbb{C}}$. Rational interpolation is a powerful method: Almost every reduced LTI system (5.2) can be obtained via rational interpolation from (5.1), see [33].

A power series expansion around $s_0 \in \mathbb{C} \setminus \lambda(A)$ with $\|(s - s_0)(A - s_0I_n)^{-1}\| < 1$

yields

$$G(s) = \sum_{j=0}^{\infty} m_j(s_0)(s - s_0)^j$$

with the so-called moments

$$m_j(s_0) = -C(A - s_0 I_n)^{-(j+1)} B = \frac{(-1)^j}{j!} \frac{\mathrm{d}^j}{\mathrm{d}s^j} G(s) \bigg|_{s=s_0}.$$

If either

$$\{(A - s_0 I_n)^{-1} B, \ldots, (A - s_0 I_n)^{-k} B\} \subseteq \operatorname{span}(V) \tag{5.5}$$

$$\text{or} \qquad \{(A^{\mathsf{T}} - \overline{s_0} I_n)^{-1} C^{\mathsf{T}}, \ldots, (A^{\mathsf{T}} - \overline{s_0} I_n)^{-k} C^{\mathsf{T}}\} \subseteq \operatorname{span}(W) \tag{5.6}$$

then the first k moments around s_0 are matched, i.e. $m_j(s_0) = \hat{m}_j(s_0)$ for $j = 0, \ldots, k-1$. If both conditions (5.5) and (5.6) are fulfilled, then even the first $2k$ moments around s_0 are matched.

For the expansion point $s_0 = \infty$ and $\|s^{-1}A\| < 1$ we use the power series expansion

$$G(s) = \sum_{j=1}^{\infty} m_j(\infty) s^{-j}$$

with the Markov parameters $m_j(\infty) = CA^{j-1}B$. If

$$\{B, AB, \ldots, A^{k-1}B\} \subseteq \operatorname{span}(V) \tag{5.7}$$

$$\text{or} \qquad \{C^{\mathsf{T}}, A^{\mathsf{T}}C^{\mathsf{T}}, \ldots, (A^{\mathsf{T}})^{k-1}C^{\mathsf{T}}\} \subseteq \operatorname{span}(W) \tag{5.8}$$

then the first k Markov parameters are matched, i.e. $m_j(\infty) = \hat{m}_j(\infty)$ for $j = 1, \ldots, k$. If both conditions (5.7) and (5.8) are fulfilled, then even the first $2k$ Markov parameters are matched.

The projection matrices can be kept real when the interpolation points occur in con-

jugated pairs as

$$\begin{aligned}&\operatorname{span}\Big((A - s_0 I_n)^{-1} v, (A - \overline{s_0} I_n)^{-1} v\Big)\\ =&\operatorname{span}\Big(\operatorname{Re}((A - s_0 I_n)^{-1} v), \operatorname{Im}((A - s_0 I_n)^{-1} v)\Big)\end{aligned}$$

holds for real vectors v.

Of course combinations of the cases mentioned above and different expansion points are possible. To obtain a well approximating reduced system the choice of the expansion points is essential and many strategies exist to obtain them, see e.g. [5, Sec. 2.2.2].

6. Numerical quadrature

We present numerical quadrature methods which are used in the third part of this work for the approximation of the Gramian. Consider an ordinary differential equation (ODE) of the type

$$\frac{\mathrm{d}}{\mathrm{d}t} y(t) = f(t, y(t)), \quad y(0) = y_0, \tag{6.1}$$

where $f\colon \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ is a given function and $y_0 \in \mathbb{R}^n$ is a given initial value. One is interested in computing the function $y\colon \mathbb{R} \to \mathbb{R}^n$ in the interval $[0, t_{\mathrm{end}}]$. There exist numerous methods for its numerical solution, see, e.g., [40, 41]. For our purposes single-step methods are sufficient. In order to compute approximate solutions $y_j \approx y(t_j)$ these methods make use of the fact that

$$y(t_j) = y(t_{j-1}) + \int_{t_{j-1}}^{t_j} f(t, y(t)) \,\mathrm{d}t$$

holds for $j = 1, 2, \ldots, N$ and time steps $t_0 = 0 < t_1 < t_2 < \cdots < t_N = t_{\mathrm{end}}$.

In particular we consider s-stage Runge-Kutta methods (see, e.g., [25, 39, 40, 41])

which are defined via

$$y_j = y_{j-1} + \omega_j \sum_{i=1}^{s} \beta_i k_i^{(j)}, \qquad j = 1, \dots, N, \tag{6.2}$$

$$k_i^{(j)} = f\Big(t_{j-1} + \gamma_i \omega_j,\, y_{j-1} + \omega_j \sum_{\ell=1}^{s} \lambda_{i\ell} k_\ell^{(j)}\Big), \qquad i = 1, \dots, s, \tag{6.3}$$

for certain $\beta_i \in \mathbb{C}$, $\gamma_i \in \mathbb{R}$, $i = 1, \dots, s$, $\lambda_{i\ell} \in \mathbb{C}$, $i, \ell = 1, \dots, s$ and time step sizes $\omega_j := t_j - t_{j-1} > 0$, $j = 1, \dots, N$. Please note that we allow for complex-valued $\lambda_{i\ell}$ and β_i unlike the standard definition of Runge-Kutta methods for the solution of (6.1). Often Runge-Kutta methods are given in short hand by the so-called Butcher tableau

$$\begin{array}{c|c} \gamma & \Lambda \\ \hline & \beta^{\mathsf{T}} \end{array} = \begin{array}{c|cccc} \gamma_1 & \lambda_{11} & \lambda_{12} & \dots & \lambda_{1s} \\ \gamma_2 & \lambda_{21} & \lambda_{22} & \dots & \lambda_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_s & \lambda_{s1} & \lambda_{s2} & \dots & \lambda_{ss} \\ \hline & \beta_1 & \beta_2 & \dots & \beta_s \end{array} \tag{6.4}$$

with $\Lambda \in \mathbb{C}^{s \times s}$, $\beta \in \mathbb{C}^s$ and $\gamma \in \mathbb{R}^s$.

The most involved part in the iteration is the calculation of $k_i^{(j)}$ in (6.3). If in the Butcher tableau Λ is a strict lower triangular matrix, then the $k_i^{(j)}$ can be calculated explicitly one after another and the resulting method is called an explicit Runge-Kutta method. Otherwise they are only defined implicitly and a system of (in general nonlinear) equations with sn unknowns has to be solved to obtain them. One strategy to simplify the computation is by using lower triangular matrices Λ, resulting in so-called diagonally implicit Runge-Kutta (DIRK) methods [46]. Then the system of equations is uncoupled into a sequence of s systems. Another kind of methods, derived from DIRK methods, are the Rosenbrock-Wanner methods [41, Ch. IV.7]. There, the nonlinear function f is approximated by a linear function. If the function f to be integrated is linear, then the Rosenbrock-Wanner methods coincide with Runge-Kutta methods.

The function

$$R(z) = 1 + z\beta^{\mathsf{T}}(I_s - z\Lambda)^{-1}\mathbb{1}_s = \frac{\det(I_s - z(\Lambda - \mathbb{1}_s\beta^{\mathsf{T}}))}{\det(I_s - z\Lambda)} \tag{6.5}$$

is called the *stability function* of the Runge-Kutta method given by (6.4). When a Runge-Kutta method is applied to the linear differential equation $y' = \lambda y$ the iteration is given by $y_k = R(z)y_{k-1}$ with $z = \omega\lambda$. The method is said to be *A-stable* if all z with $\mathrm{Re}(z) < 0$ are in the region of absolute stability, that is the set of all $z = \omega\lambda$ with $|R(z)| < 1$. For explicit Runge-Kutta methods the region of absolute stability is small and bounded. On the other hand, implicit Runge-Kutta methods have much larger regions of absolute stability and the time step size can be chosen based on the desired accuracy, not depending on stability constraints.

Part II.

Efficient solution of large-scale Riccati equations

7. Introduction

In this part the continuous-time algebraic Riccati equation

$$\begin{aligned} 0 = \mathcal{R}(X) &:= A^*X + XA + C^*C - XBB^*X \\ &= (A^* - XBB^*)X + X(A - BB^*X) + C^*C + XBB^*X \end{aligned} \tag{1.1 revisited}$$

with complex system matrices $A \in \mathbb{C}^{n\times n}$ large and sparse and $B \in \mathbb{C}^{n\times m}$, $C \in \mathbb{C}^{p\times n}$ is considered. The solution of this Riccati equation is necessary e.g. in optimal control for the LTI system (5.1), where an input function of the form $u(t) = -Fx(t)$ with $F \in \mathbb{C}^{m\times n}$ is wanted which minimizes the cost functional

$$\int_0^\infty \big(y(t)^*y(t) + u(t)^*u(t)\big)\,\mathrm{d}t.$$

One finds that the optimal control is given by $u(t) = -B^*Xx(t)$, i.e. for $F = B^*X$, where X is a solution of the Riccati equation (1.1), see [50, Ch. 16].

When (A, B) is stabilizable and (C, A) is observable, then the stabilizing solution X of the Riccati equation exists and is unique, cf. Section 2.5. As the direct solution of the Riccati equation is computationally infeasible for large dimensions n, we are interested in an approximate solution of the form $\tilde{X} = ZYZ^* \approx X$ where Z is a rectangular matrix with only few columns and Y is a small square Hermitian matrix. This form of approximation gives two degrees of freedom: The approximation space, that is, the space spanned by the columns of Z, and the choice of the approximate solution in this space which is determined by Y. Basically, there are three families of methods producing such

a low rank approximation [8], which are described briefly next. All these methods have in common that (rational) Krylov subspaces are used as approximation spaces.

Low rank Newton-Kleinman methods treat (1.1) as a nonlinear equation and apply Newton's method, so in each iteration step a linear matrix equation has to be solved, see e.g. [11] and the references therein. Recently, the projected Newton-Kleinman method was derived in [57] which is a projection scheme where inner linear solves are performed only implicitly, making this approach competitive to the other methods mentioned here.

In projection methods, such as the extended Krylov subspace method [42] and the rational Krylov subspace method [64, 62] (denoted **KSM*), the factor Z is chosen such that its columns are an orthonormal basis for a (rational) Krylov subspace. The matrix Y is chosen as the solution of an orthogonal projection of the Riccati equation onto the selected Krylov subspace. In *KSM orthonormalization of the Krylov basis is necessary and only orthogonal projections are employed.

The third family are the ADI-type methods. They are well known from linear matrix equations as illustrated in Section 4. Due to the nonlinearity of (1.1) the approaches from Lyapunov equations are not directly applicable for Riccati equations. The ADI-type methods for the Riccati equation include e.g. the qADI algorithm [71, 70], the algorithm of Amodei and Buchot [2], the RADI iteration [9] and the Cayley subspace iteration [52]. In fact all these methods produce (at least theoretically) the same approximate solution, which was shown laboriously e.g. in [7, 9]. The term *Riccati ADI methods* (proposed in [9]) will be used throughout this work to denote the ADI-type methods for Riccati equations which generate the *ADI approximate solution*. The approximation space in the Riccati ADI methods turns out to be a rational Krylov subspace. In contrast to the other two families this space is not chosen directly but determined by the used shift parameters, see [9, Prop. 2]. Another important property of the ADI approximate solution is that the residual matrix is of rank p, see e.g. [9, Prop. 1]. Due to its symmetry and positive definiteness the residual can thus be factorized into $\mathcal{R}(\tilde{X}) = \tilde{R}\tilde{R}^*$ with the *Riccati residual factor* $\tilde{R} \in \mathbb{C}^{n\times p}$.

As discussed in [9, Sec. 5], the RADI iteration is the computationally most efficient

Riccati ADI method so far. It has been formulated for the case of real system matrices A, B and C only, can handle generalized Riccati equations and provides a low rank formulation ZZ^* for the approximate solution of (1.1). The method offers a shift strategy for fast convergence and techniques to reduce the use of complex arithmetic in case of complex shift parameters. In each iteration step the Sherman-Morrison-Woodbury (SMW) formula [35, Ch. 2.1.4] is used to solve the occurring linear system, which becomes expensive in case of a system matrix B with many columns, i.e. with large m.

The approach of this work is as follows. An approximate solution of the Riccati equation (1.1) with the structure $X_j = Z_j Y_j Z_j^*$ is sought where Z_j is a basis of a (rational) Krylov subspace $\mathcal{K}_j(A^*, C^*, s)$ (cf. Section 3). The Krylov basis is considered independently from the calculation of the small matrix Y_j. We obtain two different kinds of methods which are determined by a rank and a projection condition imposed on the Riccati residual $\mathcal{R}(X_j)$.

For the first type of methods we require that the Riccati residual is of rank p. This rank condition seems arbitrary. It is justified by the following observation: In the ADI-type methods the approximation space is a rational Krylov subspace with specific poles and the Riccati residual is of rank p. We prove that these two conditions already guarantee existence and uniqueness of such an approximate solution. Hence a rank-p residual solution is an ADI approximate solution, even when it is not derived using an alternating direction approach. Two new efficient iterative methods, which generate the same approximate solution as the other ADI-type methods, are derived which differ in the way the Krylov subspace is expanded. It is notable that in these methods no orthonormalization of the involved Krylov basis is necessary. The derived methods can handle complex and generalized Riccati equations and provide realification in case of real system matrices but complex shifts. Further, a way to parallelize the solution of the appearing large-scale linear systems in both algorithms is presented, which speeds up the most expensive part of the iterations.

In the methods of the second kind the Riccati residual is projected to a Krylov subspace. The resulting projected Riccati equation is then solved for Y_j. The considered

projections do not need to be orthogonal and are built from the matrices appearing in the used rational Arnoldi decompositions. As projection methods are generalized from orthogonal projections to oblique projections we called this type of method *general projection method (GPM)*. We present a way to efficiently evaluate the norm of the residual and prove that a certain truncated approximate solution can be interpreted as the solution of the Riccati residual projected to a subspace of the Krylov subspace spanned by Z_j. This allows compression of the approximate solution and gives us a way to efficiently evaluate the norm of the resulting residual.

Summing up, the contribution of this part is the development of computationally more efficient rank-p residual methods and the proof that these methods are equivalent to ADI methods. Further, projection methods are generalized to oblique projections and truncated projection solutions are connected with projections onto subspaces of Krylov subspaces. We rely on efficient shift strategies from the literature to generate a good approximation space, which is crucial for a good approximation and fast convergence, see, e.g., [9, Sec. 4.5] and the references therein.

The rest of this part is organized as follows. A reformulation of the residual in terms of a rational Arnoldi decomposition which is used throughout this part is derived in Section 7.1. In Section 8 we present our existence and uniqueness result for the approximate solution with a rank-p residual. The derivation of the two iterative methods generating an ADI approximate solution is given in Section 9. In Section 10 we indicate how to deal with generalized Riccati equations and show how our method simplifies in case of linear matrix equations. The general projection method is introduced in Section 11. In the numerical experiments in Section 12 we measure the performance of our new iterative ADI-type methods, demonstrate the effects of parallelization, compare projection methods with different projection directions and show the effect of truncation of the approximate solution. Concluding remarks are given in Section 13.

The ADI related findings in this part, i.e. Sections 7.1 and 8 to 10, as well as the numerical experiments in Sections 12.1 and 12.2, have been published in [22]. For this work Section 8 has been restructured. Further, the derivation of our ADI-type algorithms in

Section 9 was simplified: Here the approximate solution has the structure $X_j = Z_j Y_j Z_j^*$ with an arbitrary Y_j, while in [22] such an approximate solution was constructed with $Y_j = I$. The general projection method in Section 11 and the corresponding numerical experiments in Sections 12.3 and 12.4 have not been published at the time of this publication.

7.1. Residual reformulation

The foundation of all subsequent results is the reformulation of the Riccati residual in terms of a rational Arnoldi decomposition. To this end let

$$A^* V_{j+1} \underline{K_j} = V_{j+1} \underline{H_j} \tag{7.1}$$

be a BRAD associated to the Krylov subspace $\mathcal{K}_j := \mathcal{K}_j(A^*, C^*, s)$. For our purpose it is important to distinguish between the Krylov subspace $\mathcal{K}_j$ with basis $Z_j = V_{j+1}\underline{K_j} \in \mathbb{C}^{n\times jp}$ and the augmented Krylov subspace $\mathcal{K}_j^+ := \mathcal{K}_j^+(A^*, C^*, s)$ with basis $V_{j+1} \in \mathbb{C}^{n\times(j+1)p}$. The approximate solution $X_j = Z_j Y_j Z_j^*$ clearly lies in the Krylov subspace $\mathcal{K}_j$. On the other hand, the Riccati residual $\mathcal{R}(X_j)$ can only be formulated in terms of the augmented Krylov subspace $\mathcal{K}_j^+$ as we will see next.

By definition the starting (block) vector $C^* \in \mathbb{C}^{n\times p}$ is an element of the augmented Krylov subspace, but not necessarily of the Krylov subspace $\mathcal{K}_j$. In coordinates determined by the basis V_{j+1} we represent the starting vector by $v \in \mathbb{C}^{(j+1)p\times p}$ via

$$C^* = V_{j+1} v. \tag{7.2}$$

Herewith, using $Z_j = V_{j+1}\underline{K_j}$ and the RAD relation (7.1) we rewrite the Riccati residual

for $X_j = Z_j Y_j Z_j^*$ as

$$\begin{aligned}
\mathcal{R}(X_j) &= A^* X_j + X_j A + C^* C - X_j B B^* X_j \\
&= A^* Z_j Y_j Z_j^* + Z_j Y_j Z_j^* A + C^* C - Z_j Y_j Z_j^* B B^* Z_j Y_j Z_j^* \\
&= A^* V_{j+1} \underline{K_j} Y_j \underline{K_j}^* V_{j+1}^* + V_{j+1} \underline{K_j} Y_j \underline{K_j}^* V_{j+1}^* A + V_{j+1} v v^* V_{j+1}^* \\
&\quad - V_{j+1} \underline{K_j} Y_j \underbrace{\underline{K_j}^* V_{j+1}^* B B^* V_{j+1} \underline{K_j}}_{=:S_j} Y_j \underline{K_j}^* V_{j+1}^* \\
&= V_{j+1} \left(\underline{H_j} Y_j \underline{K_j}^* + \underline{K_j} Y_j \underline{H_j}^* + v v^* - \underline{K_j} Y_j S_j Y_j \underline{K_j}^* \right) V_{j+1}^*. \qquad (7.3)
\end{aligned}$$

The term in brackets is a square $(j+1)p \times (j+1)p$ matrix which specifies the Riccati residual in the coordinates given by the basis V_{j+1} of the augmented Krylov subspace $\mathcal{K}_j^+$. All conditions imposed on the residual will be taken into account in terms of this small matrix.

8. Existence and uniqueness of the Riccati ADI solution

All known Riccati ADI methods are equivalent, as stated in the introduction. They produce approximate solutions to the Riccati equation (1.1) which span a rational Krylov subspace and yield a residual of rank p. In this section we derive an existence and uniqueness result for rank-p residual solutions $X_j = Z_j Y_j Z_j^*$, where Z_j is a basis of a Krylov subspace $\mathcal{K}_j(A^*, C^*, s)$. The result implies equivalence of all methods which yield an approximate solution with a rank-p residual, which in particular includes the ADI methods. For ease of presentation we consider the case $p = 1$, i.e. $C^* \in \mathbb{C}^{n \times 1}$. We comment on the necessary modifications for a general $p \in \mathbb{N}$ at the end of this section.

Due to the structure of the approximate solution X_j all choices of a basis Z_j of the Krylov subspace are equivalent, as a transition matrix for a change of basis can be incorporated into Y_j. Note that in the context of the approximations with a rank one residual we do not assume orthonormality of any Krylov basis. Here our point of view is an analytical rather than a numerical one, meaning that in this section we state

conditions which are as general as possible to guarantee existence and uniqueness. Thus we allow the shifts to be arbitrary complex numbers, while for a good approximation one has to choose the shifts with care, see, e.g., [9, Sec. 4.5] and the references therein. It turns out that there exist many approximate solutions $X_j = Z_j Y_j Z_j^*$ which yield a rank one residual, but only one approximate solution of rank j, i.e. with a regular $Y_j \in \mathbb{C}^{j\times j}$.

Before we state our existence and uniqueness result we present two technical lemmas. The first lemma guarantees the existence of the RAD from (7.1) in the special form

$$A^* \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} 0 \\ I \end{bmatrix} = \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} h_j \\ \underline{H_{-j}} \end{bmatrix}$$

which we are going to use to prove our main result. The second lemma ensures uniqueness of the regular solution of a homogeneous Riccati equation which has to be solved to obtain our wanted approximate solution X_j with a rank one residual.

Lemma 8.1. *Let $\mathcal{K}_j(A^*, C^*, s)$ be an A^*-variant Krylov subspace with $\infty \notin s$. Then there exists a RAD*

$$A^* V_{j+1} \underline{K_j} = V_{j+1} \underline{H_j} \tag{8.1}$$

associated to this Krylov subspace with $\underline{K_j} = \left[\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right]$, $\underline{H_j} = \left[\begin{smallmatrix} h_j \\ \underline{H_{-j}} \end{smallmatrix}\right]$, an upper triangular matrix $\underline{H_{-j}}$ and $V_{j+1} = [C^*, Z_j]$ with a basis Z_j of $\mathcal{K}_j(A^*, C^*, s)$.

Proof. The Krylov subspace $\mathcal{K}_j(A^*, C^*, s)$ is A^*-variant, so the augmented Krylov subspace $\mathcal{K}_j^+(A^*, C^*, s)$ is of dimension $j+1$. Thus Theorem 3.4 guarantees the existence of a RAD

$$A^* \tilde{V}_{j+1} \underline{\tilde{K}_j} = \tilde{V}_{j+1} \underline{\tilde{H}_j} \tag{8.2}$$

associated to $\mathcal{K}_j(A^*, C^*, s)$ with starting vector $C^* \in \operatorname{span}(\tilde{V}_{j+1} e_1)$. We aim at constructing an upper triangular matrix R to transform the RAD (8.2) into the RAD (8.1) with $V_{j+1} = \tilde{V}_{j+1} R$, $\underline{K_j} = R^{-1} \underline{\tilde{K}_j}$ and $\underline{H_j} = R^{-1} \underline{\tilde{H}_j}$. This regular R is constructed as

follows. Let $\alpha \in \mathbb{C}$ so that $C^* = \tilde{V}_{j+1}\alpha e_1$ and set $R = [\alpha e_1, \underline{\tilde{K}_j}]$ to obtain $\underline{K_j} = \begin{bmatrix} 0 \\ I \end{bmatrix}$ and $V_{j+1}e_1 = C^*$. As we assumed that $\infty \notin s$ we have, due to (3.3) and (3.6), that $C^* \notin \mathcal{K}_j(A^*, C^*, s) = \operatorname{span}(\tilde{V}_{j+1}\underline{\tilde{K}_j})$. Therefore $e_1 \notin \operatorname{span}(\underline{\tilde{K}_j})$ and R is indeed regular. As $\underline{\tilde{K}_j}$ is an upper-Hessenberg matrix, R is an upper triangular matrix and so is $\underline{H_{-j}}$. Finally, the structure of $\underline{K_j}$ and (3.6) imply that $Z_j = V_{j+1}\underline{K_j}$ is a basis of $\mathcal{K}_j(A^*, C^*, s)$, which completes the proof. □

Lemma 8.2. *Let a RAD of the form* (8.1) *as in Lemma 8.1 be associated to a Krylov subspace* $\mathcal{K}_j(A^*, C^*, s)$ *with* $s \cap -\overline{s} = \emptyset$. *Then among all solutions of the homogeneous Riccati equation*

$$\underline{H_{-j}}Y_j + Y_j\underline{H_{-j}}^* - Y_j(S_j + h_j^*h_j)Y_j = 0 \tag{8.3}$$

there exists a unique full-rank solution.

Proof. A full-rank solution Y_j of (8.3) exists if and only if the equivalent Lyapunov equation

$$0 = \tilde{Y}_j\underline{H_{-j}} + \underline{H_{-j}}^*\tilde{Y}_j - (S_j + h_j^*h_j) \tag{8.4}$$

is solved by a full-rank matrix $\tilde{Y}_j$ and in that case $Y_j = \tilde{Y}_j^{-1}$ holds. Due to Theorem 3.4 the eigenvalues of $\underline{H_{-j}}$ are equal to the shifts s, so the shift condition $s \cap -\overline{s} = \emptyset$ ensures that $\underline{H_{-j}}$ and $-\underline{H_{-j}}^*$ have no eigenvalues in common. Hence the Lyapunov equation (8.4) is uniquely solvable, see Section 2.5.

We next prove that the solution $\tilde{Y}_j$ of (8.4) is regular. For this assume that $(h_j, \underline{H_{-j}})$ is observable and so $(\underline{H_{-j}}^*, h_j^*)$ is controllable [3, Thm. 4.23]. This implies controllability of $\left(\underline{H_{-j}}^*, [h_j^*, Z_j^*B]\right)$ where

$$\begin{bmatrix} h_j^* & Z_j^*B \end{bmatrix} \begin{bmatrix} h_j \\ B^*Z_j \end{bmatrix} = h_j^*h_j + S_j$$

holds with S_j from the residual formula (7.3). Herewith regularity of $\tilde{Y}_j$ can be shown

in full analogy to the proof of [50, Thm. 5.3.1 (b)].

Assume to the contrary that $(h_j, \underline{H_{-j}})$ is unobservable, i.e. the observability matrix $\left[h_j^*, \underline{H_{-j}}^* h_j^*, \ldots, (\underline{H_{-j}}^*)^{j-1} h_j^*\right]^*$ is of rank smaller than j. Then due to [3, Thm. 4.26] there exists an vector $u \neq 0$ with $h_j u = 0$ and $\underline{H_{-j}} u = \mu u$. It holds $\mu \in s$ as the eigenvalues of $\underline{H_{-j}}$ are the poles of the Krylov subspace. Thus we find due to (8.1)

$$\begin{aligned} A^* V_{j+1} \underline{H_j} u &= A^* V_{j+1} \begin{bmatrix} h_j \\ \underline{H_{-j}} \end{bmatrix} u = A^* V_{j+1} \begin{bmatrix} 0 \\ I \end{bmatrix} \mu u \\ &= \mu V_{j+1} \underline{H_j} u \end{aligned}$$

i.e. $V_{j+1}\underline{H_j}u$ is an eigenvector of A^* with eigenvalue μ. This is a contradiction to the definition of rational Krylov subspaces because the poles of the Krylov subspace must be distinct from the eigenvalues of A^*. □

Now the main result of this section is proven, which guarantees existence and uniqueness of a rank-j approximate solution which results in a rank one residual under some mild assumptions on the poles of the Krylov subspace. Hereby equivalence of all rank-p residual methods, including the ADI methods, is implied.

Theorem 8.3. *Let $\mathcal{K}_j \coloneqq \mathcal{K}_j(A^*, C^*, s)$ be an A^*-variant Krylov subspace fulfilling the shift conditions $\infty \notin s$ and $s \cap -\overline{s} = \emptyset$. Then there exists a unique rank-j approximate solution $X_j = Z_j Y_j Z_j^*$ with a basis Z_j of $\mathcal{K}_j$ so that the residual $\mathcal{R}(X_j)$ is of rank one.*

Proof. Let the RAD (8.1) associated to $\mathcal{K}_j$ with $V_{j+1} = [C^*, Z_j]$ and $\underline{K_j} = \left[\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right]$ from Lemma 8.1 be given, so we find $Z_j = V_{j+1}\underline{K_j}$ as basis for $\mathcal{K}_j$, $v = e_1$ in (7.2) and $S_j = Z_j^* B B^* Z_j$ in (7.3).

Observe that the rank of the residual $\mathcal{R}(X_j)$ is the same as the rank of the inner matrix in (7.3),

$$M_j \coloneqq \underline{H_j} Y_j \underline{K_j}^* + \underline{K_j} Y_j \underline{H_j}^* + v v^* - \underline{K_j} Y_j S_j Y_j \underline{K_j}^*,$$

as V_{j+1} is a basis of the augmented Krylov subspace $\mathcal{K}_j^+$. Hence, to obtain a rank one

residual, the matrix Y_j has to be chosen such that M_j is of rank one. With the RAD used here we find

$$\begin{aligned} M_j &= \underline{H_j} Y_j \underline{K_j}^* + \underline{K_j} Y_j \underline{H_j}^* + e_1 e_1^* - \underline{K_j} Y_j S_j Y_j \underline{K_j}^* \\ &= \begin{bmatrix} 0 & h_j Y_j \\ 0 & \underline{H_{-j}} Y_j \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ Y_j h_j^* & Y_j \underline{H_{-j}}^* \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & Y_j S_j Y_j \end{bmatrix} \\ &= \begin{bmatrix} 1 & h_j Y_j \\ Y_j h_j^* & \underline{H_{-j}} Y_j + Y_j \underline{H_{-j}}^* - Y_j S_j Y_j \end{bmatrix}. \end{aligned} \tag{8.5}$$

The upper left entry of M_j is 1, so due to the assumption that M_j is of rank one, M_j is determined by its first row and column and has a rank-1 factorization. This rank-1 factorization is given by

$$M_j = \begin{bmatrix} 1 \\ Y_j h_j^* \end{bmatrix} \begin{bmatrix} 1 & h_j Y_j \end{bmatrix} \tag{8.6}$$

$$= \begin{bmatrix} 1 & h_j Y_j \\ Y_j h_j^* & Y_j h_j^* h_j Y_j \end{bmatrix}. \tag{8.7}$$

Clearly the lower right blocks of (8.5) and (8.7) have to coincide, so Y_j must be chosen to be a solution of

$$\begin{aligned} 0 &= \underline{H_{-j}} Y_j + Y_j \underline{H_{-j}}^* - Y_j S_j Y_j - Y_j h_j^* h_j Y_j \\ &= \underline{H_{-j}} Y_j + Y_j \underline{H_{-j}}^* - Y_j (S_j + h_j^* h_j) Y_j. \end{aligned} \tag{8.8}$$

To obtain a rank-j approximate solution $X_j = Z_j Y_j Z_j^*$ the inner matrix $Y_j \in \mathbb{C}^{j \times j}$ must be regular. From Lemma 8.2 we find that the desired regular Y_j is unique and obtained as the inverse of the solution of the Lyapunov equation (8.3). □

Summing up, the rank-j solution $X_j = Z_j Y_j Z_j^*$ with a residual of rank one is obtained

by using the RAD from Lemma 8.1, where Y_j is determined via the Lyapunov equation

$$Y_j^{-1}\underline{H_{-j}} + \underline{H_{-j}}^* Y_j^{-1} - (S_j + h_j^* h_j) = 0. \tag{8.9}$$

The residual factor R_j of the rank-1 residual $\mathcal{R}(X_j) = R_j R_j^*$ is given by

$$R_j = V_{j+1} \begin{bmatrix} 1 \\ Y_j h_j^* \end{bmatrix} \tag{8.10}$$

due to the rank one factorization of M_j in (8.6) and the residual formulation (7.3).

The above uniqueness result makes it simple to prove the equivalence of different Riccati ADI methods. One essentially has to verify that the approximate solution spans a rational Krylov subspace with a set of poles $s \subset \mathbb{C}$ satisfying $s \cap -\overline{s} = \emptyset$ and that the Riccati residual is of rank p.

Remark 8.4. Under the conditions of Theorem 8.3 with $\infty \notin s$ and $s \cap -\overline{s} = \emptyset$ there exist many approximate solutions $\tilde{X}_j = Z_j \tilde{Y} Z_j^*$ of (7.3) resulting in a rank-1 residual if the rank-j condition on $\tilde{X}_j$ is omitted. Consider e.g. the trivial solution $\tilde{Y} = 0$ of (8.8) resulting in the residual factor C^*. For other approximations with rank-1 residual consider a subset $\tilde{s} \subset s$ with $\tilde{j}$ elements. Clearly the Krylov subspace $\mathcal{K}_{\tilde{j}}(A^*, C^*, \tilde{s})$ fulfills the assumptions of Theorem 8.3 and so there exists a rank-$\tilde{j}$ approximate solution $\tilde{X}_j$ resulting in a rank-1 residual. Due to $\mathcal{K}_{\tilde{j}}(A^*, C^*, \tilde{s}) \subset \mathcal{K}_j(A^*, C^*, s)$ this solution can be represented by $\tilde{X}_j = Z_j \tilde{Y} Z_j^*$ with a rank-$\tilde{j}$ matrix $\tilde{Y}$. If the shifts in s are pairwise distinct there exist 2^j subsets of s and so 2^j approximate solutions of the form $Z_j \tilde{Y} Z_j^*$ yielding a rank-1 residual can be constructed as described above. In other words, the rank-j condition on Y_j ensures that not only the space spanned by the solution factor Z_j but also by the approximate solution X_j is the Krylov subspace $\mathcal{K}_j(A^*, C^*, s)$.

We next show that the shift condition $\infty \notin s$ is necessary for the existence of a rank-j approximate solution with a rank one residual.

Theorem 8.5. *Let $\mathcal{K}_j \coloneqq \mathcal{K}_j(A^*, C^*, s)$ be an A^*-variant Krylov subspace with $\infty \in s$. Then no rank-j approximate solution $X_j = Z_j Y_j Z_j^*$ with a basis Z_j of $\mathcal{K}_j$ exists so that*

the residual $\mathcal{R}(X_j)$ is of rank one.

Proof. As in the proof of Lemma 8.1 we can transform a RAD $A^*\tilde{V}_{j+1}\underline{\tilde{K}_j} = \tilde{V}_{j+1}\underline{\tilde{H}_j}$ associated to $\mathcal{K}_j(A^*, C^*, s)$ with a regular matrix $R = [w, \underline{\tilde{K}_j}]$ to obtain a RAD with $\underline{K_j}$ and $\underline{H_j}$ as in (8.1), but with a different V_{j+1}. Due to $\infty \in s$ we find $C^* \in \mathcal{K}_j(A^*, C^*, s) = \operatorname{span}(V_{j+1}\underline{K_j})$, so there is a vector $\tilde{v} \in \mathbb{C}^j$ with $C^* = V_{j+1}\underline{K_j}\tilde{v}$. Thus we have $v = \left[\begin{smallmatrix} 0 \\ \tilde{v} \end{smallmatrix}\right]$ in (7.2) because of the special structure of $\underline{K_j}$. Again from (7.3) we find that the rank of the residual is the same as the rank of the inner matrix

$$\begin{aligned} M_j &= \underline{H_j} Y_j \underline{K_j}^* + \underline{K_j} Y_j \underline{H_j}^* + vv^* - \underline{K_j} Y_j S_j Y_j \underline{K_j}^* \\ &= \begin{bmatrix} 0 & h_j Y_j \\ 0 & \underline{H_{-j}} Y_j \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ Y_j h_j^* & Y_j \underline{H_{-j}}^* \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & \tilde{v}\tilde{v}^* \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & Y_j S_j Y_j \end{bmatrix} \\ &= \begin{bmatrix} 0 & h_j Y_j \\ Y_j h_j^* & \tilde{v}\tilde{v}^* + \underline{H_{-j}} Y_j + Y_j \underline{H_{-j}}^* - Y_j S_j Y_j \end{bmatrix}. \end{aligned}$$

Assume this matrix is of rank one. Then due to the first diagonal entry being zero the first row and column have to be zero, i.e. $Y_j h_j^* = 0$. This is only possible if Y_j is singular or $h_j = 0$. In the latter case the Krylov subspace is A^*-invariant. In both cases the assumptions of this theorem are violated and no approximation with the desired properties exists. □

Now the general case of an arbitrary $p \in \mathbb{N}$, i.e. $C^* \in \mathbb{C}^{n\times p}$, is discussed. We aim at constructing an approximate solution $X_j = Z_j Y_j Z_j^*$ of the Riccati equation (1.1) where the residual $\mathcal{R}(X_j) = R_j R_j^*$ is of rank p and with $Z_j \in \mathbb{C}^{n\times jp}$ and $Y_j \in \mathbb{C}^{jp\times jp}$ regular. The residual factor $R_j \in \mathbb{C}^{n\times p}$ lies in a block rational Krylov subspace (3.9) instead of a Krylov subspace (3.2). To obtain a rank-p residual we proceed in full analogy to the rank-1 residual essentially by replacing vectors with block vectors (i.e. matrices with p columns) and scalars by scalar p-by-p matrices (i.e. multiples of the p-dimensional identity matrix). We briefly comment on the changes to be made to obtain the sought-after approximation. Let a block rational Krylov subspace $\mathcal{K}_j^{\square}(A^*, C^*, s)$ be given with

shifts $s = \{\mu_1, \ldots, \mu_j\} \subset \mathbb{C}$ fulfilling $s \cap -\overline{s} = \emptyset$. Let

$$A^* V_{j+1} \underline{K_j} = V_{j+1} \underline{H_j}$$

be an associated BRAD as defined in Definition 3.7 with $V_{j+1} = [C^*, Z_j] \in \mathbb{C}^{n \times jp}$ of full rank and block upper-Hessenberg matrices

$$\underline{K_j} = \begin{bmatrix} 0 \\ I \end{bmatrix} \in \mathbb{R}^{(j+1)p \times jp}$$

$$\text{and } \underline{H_j} = \begin{bmatrix} h_j \\ \underline{H_{-j}} \end{bmatrix} \in \mathbb{C}^{(j+1)p \times jp}, \text{ with } h_j \in \mathbb{C}^{p \times jp}, \underline{H_{-j}} \in \mathbb{C}^{jp \times jp}$$

in analogy to the special RAD (8.1). In accordance to property 3 of Definition 3.7 the relation $H_{i+1,i} = \mu_i I_p$ must hold for the subdiagonal blocks of $\underline{H_j}$, which are the diagonal blocks of its square lower submatrix $\underline{H_{-j}}$. With the above BRAD the residual equation (7.3) holds with the block vector $v = \left[\begin{smallmatrix} I_p \\ 0 \end{smallmatrix}\right]$ and the Lyapunov equation (8.9) to determine Y_j can be derived in full analogy. The resulting approximation $X_j = Z_j Y_j Z_j^*$ yields a rank-p residual.

9. Two new iterative Riccati ADI methods

In the previous section the rank-p residual approximate solution $X_j = Z_j Y_j Z_j^*$ of the Riccati equation (1.1) is obtained in two steps: First, a Krylov basis Z_j with a corresponding RAD is generated, then a small-scale Lyapunov equation is solved for Y_j^{-1}. In this section these two steps are combined in an iterative way. The basis of the Krylov subspace is expanded incrementally, simultaneously the solution of the small-scale Lyapunov equation is updated. Although we do not pursue an alternating direction approach we stick to the term Riccati ADI method, as our approach and the ADI methods are equivalent.

Let a block Krylov subspace $\mathcal{K}_j(A^*, C^*, s)$ with an associated BRAD as in Lemma 8.1

be given, i.e.

$$A^* \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} 0 \\ I \end{bmatrix} = \begin{bmatrix} C^* & Z_j \end{bmatrix} \underbrace{\begin{bmatrix} h_j \\ \underline{H_{-j}} \end{bmatrix}}_{=\underline{H_j}}. \tag{9.1}$$

Then the rank-p residual approximation is given by $X_j = Z_j Y_j Z_j^*$ where Y_j is determined by the Lyapunov equation (8.9). Assume that the Krylov subspace is expanded by adding one or several new poles, resulting in the corresponding expanded BRAD

$$A^* \begin{bmatrix} C^* & Z_j & \tilde{Z} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ I & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} C^* & Z_j & \tilde{Z} \end{bmatrix} \underbrace{\begin{bmatrix} h_j & U_1 \\ \underline{H_{-j}} & U_2 \\ 0 & D \end{bmatrix}}_{=\underline{H_{j+1}}}. \tag{9.2}$$

To obtain the approximate solution $X_{j+1} = Z_{j+1} Y_{j+1} Z_{j+1}^*$ with a rank-p residual we choose Y_{j+1} so that its inverse Y_{j+1}^{-1} is the solution of the Lyapunov equation (8.9). We set $Z_{j+1} = [Z_j, \tilde{Z}]$, $h_{j+1} = [h_j, U_1]$, $\underline{H_{-(j+1)}} = \left[\begin{smallmatrix} \underline{H_{-j}} & U_2 \\ 0 & D \end{smallmatrix}\right]$ from the expanded BRAD and partition $Y_{j+1}^{-1} = \left[\begin{smallmatrix} Y_{11} & Y_{12} \\ Y_{12}^* & Y_{22} \end{smallmatrix}\right]$. Then the Lyapunov equation (8.9) reads

$$\begin{aligned} 0 = & \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{12}^* & Y_{22} \end{bmatrix} \begin{bmatrix} \underline{H_{-j}} & U_2 \\ 0 & D \end{bmatrix} + \begin{bmatrix} \underline{H_{-j}}^* & 0 \\ U_2^* & D^* \end{bmatrix} \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{12}^* & Y_{22} \end{bmatrix} \\ & - \begin{bmatrix} Z_j^* \\ \tilde{Z}^* \end{bmatrix} BB^* \begin{bmatrix} Z_j & \tilde{Z} \end{bmatrix} - \begin{bmatrix} h_j^* \\ U_1^* \end{bmatrix} \begin{bmatrix} h_j & U_1 \end{bmatrix} \\ = & \begin{bmatrix} Y_{11}\underline{H_{-j}} & Y_{11}U_2 + Y_{12}D \\ Y_{12}^*\underline{H_{-j}} & Y_{12}^*U_2 + Y_{22}D \end{bmatrix} + \begin{bmatrix} \underline{H_{-j}}^* Y_{11} & \underline{H_{-j}}^* Y_{12} \\ U_2^* Y_{11} + D^* Y_{12}^* & U_2^* Y_{12} + D^* Y_{22} \end{bmatrix} \\ & - \begin{bmatrix} Z_j^* BB^* Z_j & Z_j^* BB^* \tilde{Z} \\ \tilde{Z}^* BB^* Z_j & \tilde{Z}^* BB^* \tilde{Z} \end{bmatrix} - \begin{bmatrix} h_j^* h_j & h_j^* U_1 \\ U_1^* h_j & U_1^* U_1 \end{bmatrix}. \end{aligned} \tag{9.3}$$

The blocks of (9.3) are examined separately to obtain Y_{11}, Y_{12} and Y_{22}. The upper left

block of (9.3) yields

$$0 = Y_{11}\underline{H_{-j}} + \underline{H_{-j}}^* Y_{11} - Z_j^* BB^* Z_j - h_j^* h_j,$$

which is the Lyapunov equation (8.9). Its unique solution is given by Y_j^{-1}, so $Y_{11} = Y_j^{-1}$ holds. From the upper right block we obtain the Sylvester equation

$$0 = Y_j^{-1} U_2 + Y_{12} D + \underline{H_{-j}}^* Y_{12} - Z_j^* BB^* \tilde{Z} - h_j^* U_1 \tag{9.4}$$

which is solved for Y_{12}. The lower left block is equal to the conjugated transposed of the upper right block. From the lower right block we find that Y_{22} is the solution of the Lyapunov equation

$$0 = Y_{12}^* U_2 + Y_{22} D + U_2^* Y_{12} + D^* Y_{22} - \tilde{Z}^* BB^* \tilde{Z} - U_1^* U_1. \tag{9.5}$$

Summing up, to obtain Y_{j+1}^{-1} the previous solution Y_j^{-1} of (8.9) can be reused as $Y_{11} = Y_j^{-1}$ holds for the upper left part. Only the Sylvester equation (9.4) and the Lyapunov equation (9.5) have to be solved for Y_{12} and Y_{22} respectively. In the next subsections the calculation of the expansion matrix $\tilde{Z}$ corresponding to a new pole μ is described. It is obtained through the solution of linear systems with the residual factor on the right-hand side. By using different types of system matrices we obtain two iterative procedures: In the *Riccati RAD iteration* linear systems with matrices of the form $A^* - \mu I_n$ are solved, while in the *Lyapunov RADI iteration* systems are solved with matrices of the form $A^* - X_j BB^* - \mu I_n$. In Section 9.3 we describe how the involved quantities can be kept real in case of real system matrices and complex shifts and how we can add multiple poles by making use of the solution of independent linear systems, which can be solved in parallel. Both, realification and parallelization, are based on the flexibility that lies in the BRAD expansion with $\tilde{Z}$.

9.1. The Riccati RAD iteration

For the Riccati RAD iteration (R^2ADi) we expand the BRAD (9.1) with the block vector $\tilde{Z} = (A^* - \mu I_n)^{-1} R_j$. Rewriting this term we obtain the equivalent expression

$$A^*\tilde{Z} = R_j + \mu\tilde{Z} = \begin{bmatrix} C^* & Z_j & \tilde{Z} \end{bmatrix} \begin{bmatrix} I_p \\ Y_j h_j^* \\ \mu I_p \end{bmatrix} \tag{9.6}$$

due to the representation of the residual factor $R_j = [C^*, Z_j] \begin{bmatrix} I_p \\ Y_j h_j^* \end{bmatrix}$ from (8.10). Thus in the expanded BRAD (9.2) it holds $U_1 = I_p$, $U_2 = Y_j h_j^* U_1$ and $D = \mu I_p$. We keep using the variables U_1, U_2 and D here to keep the derivation of our iterative procedure as general as necessary for the introduction of realification and parallelization later on. We find from Definition 3.7 that the BRAD (9.1) is expanded with the pole μ since $D = \mu I_p$ holds.

To obtain Y_{12} we have to solve the Sylvester equation (9.4) which simplifies to

$$0 = Y_{12} D + \underline{H_{-j}}^* Y_{12} - Z_j^* B B^* \tilde{Z}.$$

For the calculation of Y_{22} we solve the Lyapunov equation (9.5) which becomes

$$0 = Y_{12}^* Y_j h_j^* U_1 + Y_{22} D + U_1^* h_j Y_j Y_{12} + D^* Y_{22} - \tilde{Z}^* B B^* \tilde{Z} - U_1^* U_1$$

due to the choice of U_2. The residual factor is given as in (8.10) via

$$R_{j+1} = \begin{bmatrix} C^* & Z_{j+1} \end{bmatrix} \begin{bmatrix} I_p \\ Y_{j+1} h_{j+1}^* \end{bmatrix}.$$

This iterative procedure is summarized in Algorithm 9.1. Note that to calculate $Y_j h_j^*$ a linear system of equations has to be solved as only Y_j^{-1} is available.

Remark 9.1. For a stable system matrix A shifts with positive real part have to be chosen for convergence, i.e. $s \subset \mathbb{C}_+$. This choice of shifts implies that the conditions

Algorithm 9.1 Riccati RAD iteration (R^2ADi)

Input: system matrices A, B, C, set of shifts $s \subset \mathbb{C}$ with $s \cap -\overline{s} = \emptyset$
Output: approximate solution $Z_j Y_j Z_j^*$, residual factor R_j

1: initialize $Z_0 = [\,]$, $Y_0^{-1} = [\,]$, $R_0 = C^*$, $h_0 = [\,]$, $\underline{H_{-0}} = [\,]$, $s_0 = [\,]$, $j = 0$
2: **while** not converged **do**
3: obtain new shift(s) μ from s
4: expand RAD, obtain $\tilde{Z}$, U_1, D ▷ solve $\tilde{Z} = (A^* - \mu I_n)^{-1} R_j$
5: solve $Y_{12} D + \underline{H_{-j}}^* Y_{12} - s_j^* (B^* \tilde{Z}) = 0$ for Y_{12}
6: solve $Y_{12}^* Y_j h_j^* U_1 + Y_{22} D + U_1^* h_j Y_j Y_{12} + D^* Y_{22} - \tilde{Z}^* B B^* \tilde{Z} - U_1^* U_1 = 0$ for Y_{22}
7: update $Z_{j+1} = [Z_j,\, \tilde{Z}]$, $Y_{j+1}^{-1} = \begin{bmatrix} Y_j^{-1} & Y_{12} \\ Y_{12}^* & Y_{22} \end{bmatrix}$
8: update $R_{j+1} = C^* + Z_{j+1} Y_{j+1} h_{j+1}^*$
9: update $h_{j+1} = [h_j, U_1]$, $\underline{H_{-(j+1)}} = \left[\begin{smallmatrix} H_{-j} & Y_j h_j^* U_1 \\ 0 & D \end{smallmatrix}\right]$
10: update $s_{j+1} = [s_j,\, B^* \tilde{Z}]$
11: $j = j + 1$
12: **end while**

$s \cap -\overline{s} = \emptyset$ and $\infty \notin s$ for existence and uniqueness in Theorem 8.3 are satisfied. Further the eigenvalues of $\underline{H_{-j}}$ are just the shifts, so the solution Y_j^{-1} of the Lyapunov equation (8.9) is positive definite. The Cholesky factor G_j of the Cholesky decomposition $Y_j^{-1} = G_j^* G_j$ can be updated during the iteration via

$$G_{j+1} = \begin{bmatrix} G_j & G_j^{-*} Y_{12} \\ 0 & G_{22} \end{bmatrix},$$

where G_{22} is the Cholesky factor of the Cholesky decomposition $Y_{22} - Y_{12}^* Y_j Y_{12} = G_{22}^* G_{22}$. In the approximate solution $X_j = Z_j Y_j Z_j^*$ the matrix Y_j can be eliminated via $X_j = \hat{Z}_j \hat{Z}_j$ with $\hat{Z}_j = Z_j G_j^{-1}$.

9.2. The Lyapunov RADI iteration

Now consider expansion of the BRAD (9.1) with $\tilde{Z} = (A^* - X_j BB^* - \mu I_n)^{-1} R_j$. We find equivalently

$$\begin{aligned} A^*\tilde{Z} &= R_j + X_j BB^*\tilde{Z} + \mu\tilde{Z} \\ &= \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} I_p \\ Y_j h_j^* \end{bmatrix} + Z_j Y_j Z_j^* BB^*\tilde{Z} + \mu\tilde{Z} \\ &= \begin{bmatrix} C^* & Z_j & \tilde{Z} \end{bmatrix} \begin{bmatrix} I_p \\ Y_j h_j^* + Y_j Z_j^* BB^*\tilde{Z} \\ \mu I_p \end{bmatrix} \end{aligned}$$

with the residual factor $R_j = [C^*, Z_j] \begin{bmatrix} I_p \\ Y_j h_j^* \end{bmatrix}$ as in (8.10) and $X_j = Z_j Y_j Z_j^*$. Thus the expanded BRAD (9.2) holds with $U_1 = I_p$, $U_2 = Y_j h_j^* U_1 + Y_j Z_j^* BB^*\tilde{Z}$, and $D = \mu I_p$. Again we use the generic variables U_1, U_2 and D in preparation for realification and parallelization in Section 9.3. As in the previous subsection $D = \mu I_p$ holds, so the BRAD (9.1) is expanded with the pole μ here, too.

The Sylvester equation (9.4) for Y_{12} simplifies considerably and becomes

$$0 = Y_{12} D + \underline{H_{-j}}^* Y_{12},$$

due to the additional term $Y_j Z_j^* BB^*\tilde{Z}$ in U_2. Thus it holds $Y_{12} = 0$. Inserting this term into the Lyapunov equation (9.5) for the calculation of Y_{22} we find

$$0 = Y_{22} D + D^* Y_{22} - \tilde{Z}^* BB^*\tilde{Z} - U_1^* U_1. \tag{9.7}$$

Algorithm 9.2 Lyapunov RADI iteration

Input: system matrices A, B, C, set of shifts $s \subset \mathbb{C}$ with $s \cap -\overline{s} = \emptyset$
Output: approximate solution $Z_j Y_j Z_j^*$, residual factor R_j

1: initialize $Z_0 = [\,]$, $Y_0^{-1} = [\,]$, $R_0 = C^*$, $h_0 = [\,]$, $\underline{H_{-0}} = [\,]$, $s_0 = [\,]$, $K_0 = 0$, $j = 0$
2: **while** not converged **do**
3: obtain new shift(s) μ from s
4: expand RAD, obtain $\tilde{Z}$, U_1, D ▷ solve $\tilde{Z} = (A^* - K_j B^* - \mu I_n)^{-1} R_j$
5: solve $Y_{22} D + D^* Y_{22} - \tilde{Z}^* B B^* \tilde{Z} - U_1^* U_1 = 0$ for Y_{22}
6: update $Z_{j+1} = [Z_j,\, \tilde{Z}]$, $Y_{j+1}^{-1} = \begin{bmatrix} Y_j^{-1} & 0 \\ 0 & Y_{22} \end{bmatrix}$
7: update $R_{j+1} = R_j + \tilde{Z} Y_{22}^{-1} U_1^*$
8: update $K_{j+1} = K_j + \tilde{Z} Y_{22}^{-1} (\tilde{Z}^* B)$
9: update $h_{j+1} = [h_j, U_1]$, $\underline{H_{-(j+1)}} = \left[\begin{array}{c|c} \underline{H_{-j}} & Y_j h_j^* U_1 + Y_j s_j^* (B^* \tilde{Z}) \\ \hline 0 & D \end{array}\right]$
10: update $s_{j+1} = [s_j,\, B^* \tilde{Z}]$
11: $j = j + 1$
12: **end while**

The residual factor (8.10) is updated via

$$\begin{aligned} R_{j+1} &= \begin{bmatrix} C^* & Z_{j+1} \end{bmatrix} \begin{bmatrix} I_p \\ Y_{j+1} h_{j+1}^* \end{bmatrix} \\ &= \begin{bmatrix} C^* & Z_j & \tilde{Z} \end{bmatrix} \begin{bmatrix} I_p \\ Y_j h_j^* \\ Y_{22}^{-1} U_1^* \end{bmatrix} \\ &= R_j + \tilde{Z} Y_{22}^{-1} U_1^*, \end{aligned}$$

as $Y_{j+1}^{-1} = \begin{bmatrix} Y_j^{-1} & 0 \\ 0 & Y_{22} \end{bmatrix}$ is a block diagonal matrix.

Note that for the iteration in Algorithm 9.2 it is not necessary to compute and store $\underline{H_j}$, so the lines 9, 10 and the variables h_j, $\underline{H_{-j}}$, s_j can be deleted, see Algorithm A.1 in Appendix A. Nevertheless, having $\underline{H_j}$ available might be useful in some situations and only neglectable amount of memory and computational time is necessary to create it.

Remark 9.2. The procedure derived in this subsection is connected to the RADI iteration from [9] as follows. The shift parameters α_j in the RADI iteration correspond to the negative poles of our BRADs, so when the parameters are chosen as $\mu = -\alpha_j$, an

equivalent approximate solution is obtained. For $U_1 = I_p$ and $D = \mu I_p$ as above (9.7) simplifies to

$$2\,\mathrm{Re}(\mu)Y_{22} = I_p + \tilde{Z}^* B B^* \tilde{Z},$$

which is, up to constants, equivalent to line 10 of the RADI iteration [9, Alg. 1]. Due to the Lyapunov equation (9.7) and the similarity to the RADI iteration we chose the name Lyapunov RADI iteration for Algorithm 9.2. The more general Lyapunov equation (9.7) is solved because it makes the iteration more versatile: It allows for realified and parallel RAD expansions which is described in the next subsection.

Remark 9.3. The matrix $A^* - K_j B^* - \mu I_n$ with the feedback term $K_j = X_j B$ is a dense matrix in general due to the term $K_j B^*$, even when A^* is sparse, making system solves costly. However, as $K_j B^*$ is of rank m, it was proposed in [9, Sec. 4.2] to use the Sherman-Morrison-Woodbury (SMW) formula to speed up computations. The formula reads

$$(A^* - K_j B^* - \mu I_n)^{-1} R_j = L + N(I_m - B^* N)^{-1} B^* L \tag{9.8}$$

$$\begin{bmatrix} L & N \end{bmatrix} = (A^* - \mu I_n)^{-1} \begin{bmatrix} R_j & K_j \end{bmatrix}. \tag{9.9}$$

To obtain the solution of (9.8), first the sparse linear system in (9.9) is solved for L and N, then the right-hand side of (9.8) is used.

9.3. Parallel and realified expansion of the Krylov basis

So far we have expanded the BRAD (9.1) with one new pole μ and obtained the expanded BRAD (9.2) with $U_1 = I_p$, $D = \mu I_p$ and $U_2 = Y_j h_j^* U_1$ or $U_2 = Y_j h_j^* U_1 +$

Algorithm 9.3 Simple RAD expansion

Input: residual factor R_j, shift $\mu \in \mathbb{C} \setminus \lambda(A^*)$
Output: $\tilde{Z}$, U_1, D
1: solve $\tilde{Z} = (A^* - \mu I_n)^{-1} R_j$ or $\tilde{Z} = (A^* - K_j B^* - \mu I_n)^{-1} R_j$
2: $U_1 = I_p$, $D = \mu I_p$

Algorithm 9.4 Parallel RAD expansion

Input: residual factor R_j, pairwise distinct shifts $\mu_1, \ldots, \mu_l \in \mathbb{C} \setminus \lambda(A^*)$
Output: $\tilde{Z}$, U_1, D

1: solve $\tilde{Z}_i = (A^* - \mu_i I_n)^{-1} R_j$ or $\tilde{Z}_i = (A^* - K_j B^* - \mu_i I_n)^{-1} R_j$ for $i = 1, \ldots, l$
2: $\tilde{Z} = \begin{bmatrix} \tilde{Z}_1, \ldots, \tilde{Z}_l \end{bmatrix}$
3: $U_1 = [I_p, \ldots, I_p]$, $D = \operatorname{diag}(\mu_1 I_p, \ldots, \mu_l I_p)$

$Y_j Z_j^* B B^* \tilde{Z}$ respectively. The Krylov basis in the BRAD was expanded with $\tilde{Z} = (A^* - \mu I_n)^{-1} R_j$ or $\tilde{Z} = (A^* - X_j B B^* - \mu I_n)^{-1} R_j$. This simple expansion is summarized in Algorithm 9.3.

We now describe how the BRAD can be expanded in parallel in the R^{2}ADi. Hereby we mean expanding the Krylov basis with $\tilde{Z}_i = (A^* - \mu_i I_n)^{-1} R_j$ for $i = 1, \ldots, l$, corresponding to the pairwise distinct shifts $\mu_1, \ldots, \mu_l$. These linear systems are independent from each other and can thus be solved in parallel. Rewriting these linear systems we obtain

$$A^* \tilde{Z}_i = R_j + \mu_i \tilde{Z}_i = \begin{bmatrix} C^* & Z_j & \tilde{Z}_i \end{bmatrix} \begin{bmatrix} I_p \\ Y_j h_j^* \\ \mu_i I_p \end{bmatrix}$$

as in (9.6). Thus the expanded BRAD (9.2) with $\tilde{Z} = [\tilde{Z}_1, \ldots, \tilde{Z}_l]$ is given by

$$A^* \begin{bmatrix} C^* & Z_j & \tilde{Z} \end{bmatrix} \begin{bmatrix} 0 \\ I \end{bmatrix} = \begin{bmatrix} C^* & Z_j & \tilde{Z} \end{bmatrix} \begin{bmatrix} h_j & I_p & \cdots & I_p \\ \underline{H_{-j}} & Y_j h_j^* & \cdots & Y_j h_j^* \\ 0 & \mu_1 I_p & & \\ \vdots & & \ddots & \\ 0 & & & \mu_l I_p \end{bmatrix}.$$

This procedure is summarized in Algorithm 9.4. Due to Definition 3.7 and the structure of D we find that the BRAD is expanded with the poles $\mu_1, \ldots, \mu_l$. We also refer to [68, Ch. 5.4.3] where a different approach for the parallelization of the ADI iteration for Lyapunov equations was mentioned.

Algorithm 9.5 Realified RAD expansion

Input: real residual factor R_j, shifts $a+b\imath = \mu, \overline{\mu} \in \mathbb{C} \setminus \lambda(A^*)$ with $b \neq 0$
Output: $\tilde{Z}$, U_1, D

1: solve $W = (A^* - \mu I_n)^{-1} R_j$ or $W = (A^* - K_j B^* - \mu I_n)^{-1} R_j$
2: $\tilde{Z} = [\mathrm{Re}(W), \mathrm{Im}(W)]$
3: $U_1 = [I_p, 0]$, $D = \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \otimes I_p$

Next we consider a modification of R^{2}ADi such that in case of real system matrices A, B, C and two complex conjugated shifts the iterates remain real valued and use of complex arithmetic is minimized; in short, we consider realification. Let R_j have only real entries and let the BRAD (9.1) be expanded by the two conjugated shifts $\mu, \overline{\mu} \in \mathbb{C} \setminus \mathbb{R}$, $\mu = a + b\imath$, $a, b \in \mathbb{R}$. Set $W = (A^* - \mu I_n)^{-1} R_j$ and so $\overline{W} = (A^* - \overline{\mu} I_n)^{-1} R_j$. With $S = \frac{1}{2}\left[\begin{smallmatrix} 1 & -\imath \\ 1 & \imath \end{smallmatrix}\right] \otimes I_p$ we see $[W, \overline{W}]S = [\mathrm{Re}(W), \mathrm{Im}(W)]$ and $S^{-1}\left[\begin{smallmatrix} \mu I_p & \\ & \overline{\mu} I_p \end{smallmatrix}\right] S = \left[\begin{smallmatrix} aI_p & bI_p \\ -bI_p & aI_p \end{smallmatrix}\right]$. Parallel expansion of the Krylov basis with the complex block vectors W and $\overline{W}$ yields the expanded BRAD

$$A^* \begin{bmatrix} C^* & Z_j & W & \overline{W} \end{bmatrix} \begin{bmatrix} 0 \\ I \end{bmatrix} = \begin{bmatrix} C^* & Z_j & W & \overline{W} \end{bmatrix} \left[\begin{array}{ccc} h_j & I_p & I_p \\ \underline{H_{-j}} & Y_j h_j^* & Y_j h_j^* \\ 0 & \mu I_p & \\ 0 & & \overline{\mu} I_p \end{array}\right].$$

Transformation of this BRAD with the matrix $\left[\begin{smallmatrix} I & 0 \\ 0 & S \end{smallmatrix}\right]$ then yields the equivalent real BRAD

$$A^* \begin{bmatrix} C^* & Z_j & \mathrm{Re}(W) & \mathrm{Im}(W) \end{bmatrix} \begin{bmatrix} 0 \\ I \end{bmatrix} = \begin{bmatrix} C^* & Z_j & \mathrm{Re}(W) & \mathrm{Im}(W) \end{bmatrix} \left[\begin{array}{ccc} h_j & I_p & 0 \\ \underline{H_{-j}} & Y_j h_j^* & 0 \\ 0 & aI_p & bI_p \\ 0 & -bI_p & aI_p \end{array}\right], \quad (9.10)$$

where the Krylov basis is expanded with the real block vectors $\mathrm{Re}(W)$ and $\mathrm{Im}(W)$. Note that only one (complex) system solve is necessary for the expansion with the two shifts μ and $\overline{\mu}$. The realified expansion is stated in Algorithm 9.5.

Remark 9.4. Unfortunately after the realification the relation (9.10) does not fulfill property 3 of Definition 3.7. It therefore does not satisfy the definition of a BRAD anymore. We thus propose the term *quasi BRAD* in analogy to the term quasi RAD. Further, for computational reasons it is beneficial to permute the columns of $\mathrm{Re}(W)$ and $\mathrm{Im}(W)$ so that the Krylov basis is expanded by $[\mathrm{Re}(w_1), \mathrm{Im}(w_1), \ldots, \mathrm{Re}(w_p), \mathrm{Im}(w_p)]$ for $W = [w_1, \ldots, w_p]$ as then the lower right block $\left[\begin{smallmatrix} a & b \\ -b & a \end{smallmatrix}\right] \otimes I_p$ becomes $I_p \otimes \left[\begin{smallmatrix} a & b \\ -b & a \end{smallmatrix}\right]$, a block diagonal matrix with 2-by-2 blocks on the diagonal. This permutation makes $\underline{H_{-j}}$ a quasi upper triangular matrix so the Sylvester equation in line 5 of Algorithm 9.1 can be solved efficiently with established software packages.

The parallel and realified expansion was derived here only for the Riccati RAD iteration. It can be adapted for the Lyapunov RADI iteration by simply incorporating the term $K_j B^*$ in the linear systems.

10. Discussion on Riccati ADI

In this section several aspects of the Riccati ADI methods are examined. We discuss how our algorithms have to be modified to handle generalized Riccati equations in Section 10.1 and how they simplify in the special case of Lyapunov equations in Section 10.2. For good convergence the choice of the poles of the involved Krylov subspace is crucial. We rely on a method from the literature which is presented in Section 10.3 and will be used in our numerical experiments. In Section 10.4 it is shown that the Riccati ADI approximate solution is the solution of an oblique projection of the Riccati residual. Further the residual factor is characterized as a rational function.

10.1. Generalized Riccati equations

We consider two generalizations of the Riccati equation (1.1). We first describe how the generalized Riccati equation

$$A^*XE + E^*XA + C^*C - E^*XBB^*XE = 0 \tag{10.1}$$

with an additional regular system matrix $E \in \mathbb{C}^{n\times n}$ affects our iteration. This equation is not solved directly. Instead, as in [9, Sec. 4.4], the equivalent Riccati equation

$$E^{-*}A^*X + XAE^{-1} + E^{-*}C^*CE^{-1} - XBB^*X = 0 \tag{10.2}$$

is considered. It has the same structure as (1.1) where the system matrix A and the initial residual factor C^* are replaced by AE^{-1} and $E^{-*}C^*$ respectively. It can therefore be solved with the methods described in the preceding sections. In an efficient iteration inverting E is avoided by utilizing the relation

$$((AE^{-1})^* - \mu I_n)^{-1}E^{-*}R = (A^* - \mu E^*)^{-1}R$$

with a residual factor R of (10.1). This requires the following modifications in the algorithms presented. All linear systems have to be shifted by a multiple of E^* instead of I_n, i.e. the systems $(A^* - \mu E^*)^{-1}R_j$ respectively $(A^* - K_jB^* - \mu E^*)^{-1}R_j$ have to be solved. Further, the residual and feedback updates have to be multiplied with E^* to convert these quantities corresponding to (10.2) into ones corresponding to (10.1). For instance, the residual update in line 8 of Algorithm 9.1 has to be replaced by $R_{j+1} = C^* + E^*Z_{j+1}Y_{j+1}h_{j+1}^*$ and the feedback update in line 8 of Algorithm 9.2 must be replaced by $K_{j+1} = K_j + E^*\tilde{Z}Y_{22}^{-1}(\tilde{Z}^*B)$. The algorithms with these modifications are given in Appendix A as Algorithm A.2 and Algorithm A.3.

Our approach can also be applied to the nonsymmetric Riccati equation

$$A_1^*X + XA_2 + C_1^*C_2 - XB_2B_1^*X = 0 \tag{10.3}$$

with $A_i \in \mathbb{C}^{n_i\times n_i}$, $B_i \in \mathbb{C}^{n_i\times m}$, and $C_i \in \mathbb{C}^{p\times n_i}$ for $i = 1, 2$ and $X \in \mathbb{C}^{n_1\times n_2}$. It can be solved in analogy to the symmetric Riccati equation as follows. Consider the two decompositions

$$A_i^*\begin{bmatrix} C_i^* & Z_i \end{bmatrix}\underline{K_i} = \begin{bmatrix} C_i^* & Z_i \end{bmatrix}\underline{H_i} \text{ for } i = 1, 2,$$

with $\underline{K_i} = \left[\begin{smallmatrix} 0 \\ I \end{smallmatrix}\right]$ and $\underline{H_i} = \left[\begin{smallmatrix} h_i \\ \underline{H_{-i}} \end{smallmatrix}\right]$ similar to Lemma 8.1 and with the same number of columns in Z_1 and Z_2. Then in analogy to (7.3) we can rewrite the residual (10.3) for the approximate solution $X = Z_1 Y Z_2^*$ as

$$\begin{bmatrix} C_1^* & Z_1 \end{bmatrix} \left(\underline{H_1} Y \underline{K_2}^* + \underline{K_1} Y \underline{H_2}^* + \begin{bmatrix} I_p \\ 0 \end{bmatrix} \begin{bmatrix} I_p & 0 \end{bmatrix} - \underline{K_1} Y S Y \underline{K_2}^* \right) \begin{bmatrix} C_2^* & Z_2 \end{bmatrix}^*$$

due to $Z_i = [C_i^*, Z_i]\underline{K_i}$ and with $S = Z_2^* B_2 B_1^* Z_1$. The Lyapunov equation (8.9) which determines Y^{-1} so that a rank-p residual is obtained has to be adapted accordingly. It becomes the Sylvester equation

$$0 = Y^{-1} \underline{H_{-1}} + \underline{H_{-2}}^* Y^{-1} - (S + h_2^* h_1) \tag{10.4}$$

and the approximate solution of (10.3) is given by $X = Z_1 Y Z_2^*$. Of course also the derivation of the iterative procedure in Section 9 can be transferred to the nonsymmetric Riccati equation. We briefly mention where such a procedure differs from Algorithm 9.1. On the one hand, clearly two RADs have to be created instead of one, doubling the computational effort for the solution of linear systems. On the other hand, the Lyapunov equation (8.9) becomes the Sylvester equation (10.4) and is in general solved by a non-Hermitian matrix. Therefore the upper right and lower left block of the solution of (10.4) as in (9.3) are not connected via conjugated transposition and two Sylvester equations like (9.4) have to be solved. Further, the Lyapunov equation (9.5) becomes a Sylvester equation and yields a non-Hermitian solution. For parallelization and realification no further changes are necessary as both happens in the algorithms for the RAD expansion.

10.2. Linear matrix equations

All results in this work immediately transfer to Lyapunov equations which are a special case of the Riccati equation

$$A^* X + X A + C^* C - X B B^* X = 0$$

with $B = 0$ and so $S_j = Z_j^* BB^* Z = 0$. The BRAD expansion in R^{2}ADi and the Lyapunov RADI iteration becomes identical due to $K_j = X_j B = 0$, and both iterations simplify considerably. A variant of these algorithms tailored for the Lyapunov equation $AX + XA^* + BB^* = 0$ is stated in Algorithm A.4 with the matrix A in place of A^* and the variable B instead of C^*.

For the simple RAD expansion as in Algorithm 9.3 and shifts $s \subset \mathbb{C}^+$ the iteration simplifies even further. It holds $U_1 = I_p$ and $D = \mu I_p$, so the positive definite matrix Y_{22} is given via $2\,\mathrm{Re}(\mu) Y_{22} = I_p$ which implies $Y_{22}^{-1} = 2\,\mathrm{Re}(\mu) I_p$. We thus find $Y_{j+1} = \begin{bmatrix} Y_j & 0 \\ 0 & 2\,\mathrm{Re}(\mu) I_p \end{bmatrix}$. Proceeding as in Remark 9.1 we obtain Algorithm A.5.

It is notable that even for the linear Lyapunov equations the matrix Y_j is the solution of (8.8), which is a Riccati equation, i.e. a quadratic matrix equation. This was also found in [1, Sec. IV] for large-scale Sylvester equations. However, as this small-scale Riccati equation is homogeneous, the equivalent small-scale Lyapunov equation (8.9) can be solved for the inverse of the solution instead. The discussion in Section 9.2 implies that this solution is a (block) diagonal matrix.

10.3. Shift selection

For a good approximate solution the choice of the poles of the rational Krylov subspace used in the approximate solution X_j is crucial. Many shift strategies exist, but their description is beyond the scope of this work. Here we only describe, in concise form and in our notation, the *residual Hamiltonian shift strategy* from [9, Sec. 4.5.1] which we use in our numerical experiments. For a detailed discussion of this and other shift strategies we refer to [9, Sec. 4.5].

The residual Hamiltonian shift strategy makes use of the eigenvalues of the Hamiltonian matrix

$$\mathscr{H}^{\mathrm{proj}} = \begin{bmatrix} U^* \tilde{A} U & U^* B B^* U \\ U^* R_j R_j^* U & -U^* \tilde{A}^* U \end{bmatrix}$$

where $\tilde{A} = A - BB^* X_j$ holds and U is an orthonormal matrix which spans the same

space as the last l columns of Z_j, where l is a parameter of choice. Let $\hat{\lambda}$ be an eigenvalue of $\mathscr{H}^{\text{proj}}$ with the corresponding eigenvector $\left[\begin{smallmatrix}\hat{r}\\ \hat{q}\end{smallmatrix}\right]$. The next shift is chosen as $\mu = -\hat{\lambda}$ where $\hat{\lambda}$ maximizes the expression $\|\hat{q}(\hat{q}^*\hat{r})^{-1}\hat{q}^*\|$ as a heuristic for fast convergence.

10.4. Connection of Riccati ADI to projection

We now discuss how the Riccati ADI approximate solution can be interpreted as the solution of a projection of the large-scale Riccati equation (1.1) onto a Krylov subspace. Only the case $p = 1$, i.e. $C^* \in \mathbb{C}^{n\times 1}$, is considered. Let $Z, W \in \mathbb{C}^{n\times j}$ be matrices of rank j with regular Z^*W. Consider the projection

$$\Pi = Z(W^*Z)^{-1}W^* \in \mathbb{C}^{n\times n}$$

onto $\operatorname{im}(Z)$ along $\ker(W^*)$. Set $\tilde{W} \coloneqq W(Z^*W)^{-1}$, then $\Pi = Z\tilde{W}^*$ holds with $\tilde{W}^*Z = I$. Let the approximate solution $X_j \approx X$ to the Riccati equation (1.1) lie in $\operatorname{im}(Z)$ with the representation $X_j = ZY_jZ^*$. Projection of (1.1) yields the equation

$$\Pi\mathcal{R}(X_j)\Pi^* = 0 \tag{10.5}$$

which determines Y_j.

In the following theorem we show how, with the assumptions of Theorem 8.3, the Riccati ADI approximation can be obtained as the solution of the projected Riccati equation (10.5) using an oblique projection. It is a generalization of [69, Sec. 3.2], [68, Rem. 5.16], where a similar statement for the ADI iteration to solve Lyapunov equations is presented. An extended discussion on general projection methods for solving large-scale Riccati equations is given in Section 11.

Theorem 10.1. *Let $\mathcal{K}_j(A^*, C^*, s)$ be an A^*-variant Krylov subspace with shifts $s \subset \mathbb{C}$ and $s \cap -\overline{s} = \emptyset$. Let*

$$A^*V_{j+1}\begin{bmatrix}0\\ I\end{bmatrix} = V_{j+1}\begin{bmatrix}h_j\\ \underline{H_{-j}}\end{bmatrix}$$

be an associated RAD as in Lemma 8.1 with $V_{j+1} = [C^*, Z_j]$. *Let* Π *be a projection onto* $\mathrm{im}(Z_j)$ *along* $\ker(W^*)$. *If the Riccati residual factor* R_j *as in* (8.10) *is contained in the kernel of the projection* Π, *i.e.* $\tilde{W} \perp R_j$, *then the projected equation* (10.5) *is equivalent to the small-scale Riccati equation* (8.8).

Proof. From the orthogonality condition of the residual factor $R_j = V_{j+1}\left[\begin{smallmatrix} 1 \\ Y_j h_j^* \end{smallmatrix}\right]$ we find

$$0 = \tilde{W}^* R_j = \tilde{W}^* \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} 1 \\ Y_j h_j^* \end{bmatrix} = \tilde{W}^* (C^* + Z_j Y_j h_j^*),$$

or equivalently

$$\tilde{W}^* C^* = -\tilde{W}^* Z_j Y_j h_j^* = -Y_j h_j^*.$$

Herewith we obtain

$$\tilde{W}^* V_{j+1} = \tilde{W}^* \begin{bmatrix} C^* & Z_j \end{bmatrix} = \begin{bmatrix} \tilde{W}^* C^* & \tilde{W}^* Z_j \end{bmatrix} = \begin{bmatrix} -Y_j h_j^* & I \end{bmatrix}.$$

As Z_j is a basis it holds $\ker(Z_j) = \{0\}$, so with $\Pi = Z_j \tilde{W}^*$ the projected equation (10.5) is equivalent to $\tilde{W}^* \mathcal{R}(X_j) \tilde{W} = 0$. With (7.3) and (8.5) we obtain

$$\begin{aligned}
\tilde{W}^* \mathcal{R}(X_j) \tilde{W} &= \tilde{W}^* V_{j+1} \begin{bmatrix} 1 & h_j Y_j \\ Y_j h_j^* & \underline{H_{-j}} Y_j + Y_j \underline{H_{-j}}^* - Y_j S_j Y_j \end{bmatrix} V_{j+1}^* \tilde{W} \\
&= \begin{bmatrix} -Y_j h_j^* & I \end{bmatrix} \begin{bmatrix} 1 & h_j Y_j \\ Y_j h_j^* & \underline{H_{-j}} Y_j + Y_j \underline{H_{-j}}^* - Y_j S_j Y_j \end{bmatrix} \begin{bmatrix} -h_j Y_j \\ I \end{bmatrix} \\
&= Y_j h_j^* h_j Y_j - Y_j h_j^* h_j Y_j - Y_j h_j^* h_j Y_j + \underline{H_{-j}} Y_j + Y_j \underline{H_{-j}}^* - Y_j S_j Y_j \\
&= -Y_j h_j^* h_j Y_j + \underline{H_{-j}} Y_j + Y_j \underline{H_{-j}}^* - Y_j S_j Y_j.
\end{aligned}$$

This is the right-hand side of (8.8) which concludes the proof. □

If Π in the above theorem is an orthogonal projection (as used in *KSM), we have $\mathrm{im}(\Pi) \perp \ker(\Pi)$. It follows that the conditions $R_j \in \ker(\Pi)$ and $R_j \perp \mathcal{K}_j(A^*, C^*, s)$

are equivalent. Further, the approximations generated by *KSM and the Riccati ADI methods coincide.

Although $\tilde{W}$ is unknown in practice, the small projected system matrices $\tilde{W}^*A^*Z_j$ and $\tilde{W}^*(A^* - X_jBB^*)Z_j$ can be expressed in terms of parts of the RAD. This relation is established in the next lemma, which will also be useful in the proofs of the subsequent theorems.

Lemma 10.2. *Let $\mathcal{K}_j(A^*, C^*, s)$ be an A^*-variant Krylov subspace. Let*

$$A^* \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} 0 \\ I \end{bmatrix} = \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} h_j \\ \underline{H_{-j}} \end{bmatrix} \tag{10.6}$$

be an associated RAD and let Y_j be regular such that the Lyapunov equation

$$0 = Y_j^{-1}\underline{H_{-j}} + \underline{H_{-j}}^*Y_j^{-1} - (S_j + h_j^*h_j) \tag{8.9 revisited}$$

holds, cf. Theorem 8.3. Let Π be a projection onto $\operatorname{im}(Z_j)$ along $\ker(W^*)$ with $\tilde{W} \perp R_j$. Then the relations

$$\begin{aligned} \tilde{W}^*A^*Z_j &= -Y_jh_j^*h_j + \underline{H_{-j}} \\ &= -Y_j\underline{H_{-j}}^*Y_j^{-1} + Y_jS_j, \\ \tilde{W}^*(A^* - X_jBB^*)Z_j &= -Y_j\underline{H_{-j}}^*Y_j^{-1} \\ &= \underline{H_{-j}} - Y_jS_j - Y_jh_j^*h_j, \end{aligned}$$

hold.

Proof. With $\tilde{W}^*C^* = -Y_jh_j^*$ as in the proof of Theorem 10.1, with $\tilde{W}^*Z_j = I$ and by

utilizing the RAD equation (10.6) we find

$$\begin{aligned}
-Y_j h_j^* h_j + \underline{H_{-j}} &= \tilde{W}^* C^* h_j + \tilde{W}^* Z_j \underline{H_{-j}} \\
&= \tilde{W}^* \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} h_j \\ \underline{H_{-j}} \end{bmatrix} \\
&= \tilde{W}^* A^* \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} 0 \\ I \end{bmatrix} \\
&= \tilde{W}^* A^* Z_j.
\end{aligned}$$

Multiplication of (8.9) with Y_j from the left implies

$$-Y_j h_j^* h_j + \underline{H_{-j}} = -Y_j \underline{H_{-j}}^* Y_j^{-1} + Y_j S_j,$$

which concludes the first part of the proof.

For the second part consider $S_j = Z_j^* B B^* Z_j$ and again $\tilde{W}^* Z_j = I$ to obtain

$$\begin{aligned}
Y_j S_j &= \tilde{W}^* Z_j Y_j S_j \\
&= \tilde{W}^* Z_j Y_j Z_j^* B B^* Z_j \\
&= \tilde{W}^* X_j B B^* Z_j.
\end{aligned}$$

Multiplication of (8.9) with Y_j from the left yields

$$0 = \underline{H_{-j}} + Y_j \underline{H_{-j}}^* Y_j^{-1} - Y_j (S_j + h_j^* h_j).$$

Thus we have

$$\begin{aligned}
-Y_j \underline{H_{-j}}^* Y_j^{-1} &= \underline{H_{-j}} - Y_j S_j - Y_j h_j^* h_j \\
&= \tilde{W}^* A^* Z_j - \tilde{W}^* X_j B B^* Z_j \\
&= \tilde{W}^* (A^* - X_j B B^*) Z_j
\end{aligned}$$

with $-Y_j h_j^* h_j + \underline{H_{-j}} = \tilde{W}^* A^* Z_j$ from the first part. This concludes the second part of the proof. □

The residual factor R_j from (8.10) is a linear combination of the columns of V_{j+1} and so it is an element of the augmented Krylov subspace $\mathcal{K}_j^+(A^*, C^*, s)$, i.e. a rational function in A^* multiplied with C^*. We aim at specifying this rational function, which turns out to be connected to the eigenvalues of the projected matrix $\tilde{W}^* A^* Z_j$ as stated in the next theorem.

Theorem 10.3. *Let the same assumptions as in Lemma 10.2 hold. Additionally let the augmented Krylov subspace $\mathcal{K}_j^+(A^*, C^*, s)$ be A^*-variant. Let $\mathfrak{p}_j, \mathfrak{q}_j \in \mathbf{\Pi}_{(j)}$ be normalized polynomials of degree j given by*

$$\mathfrak{p}_j(x) = \prod_{i=1}^{j} \left(x - \lambda_i^{(j)}\right) \text{ and } \mathfrak{q}_j(x) = \prod_{i=1}^{j} (x - s_i)$$

with the eigenvalues $\{\lambda_1^{(j)}, \ldots, \lambda_j^{(j)}\} = \lambda(\tilde{W}^* A^* Z_j)$ and the poles $s_i \in s$ of the Krylov subspace. Then for the residual factor (8.10)

$$R_j = \mathfrak{q}_j(A^*)^{-1} \mathfrak{p}_j(A^*) C^* = \prod_{i=1}^{j} \frac{A^* - \lambda_i^{(j)} I_n}{A^* - s_i I_n} C^*$$

holds.

Proof. By construction the polynomial $\mathfrak{q}_j$ is normalized and Z_j in the RAD (10.6) is a basis of the Krylov subspace $\mathcal{K}_j(A^*, C^*, s) = \mathcal{K}_j(A^*, C^*, \mathfrak{q}_j)$. Due to the definition of Krylov subspaces (3.2) there exists a polynomial $\breve{\mathfrak{p}} \in \mathbf{\Pi}_{(j-1)}$ of degree at most $j-1$ with

$$Z_j Y_j h_j^* = \mathfrak{q}_j(A^*)^{-1} \breve{\mathfrak{p}}(A^*) C^*.$$

Thus for the residual factor (8.10)

$$\begin{aligned} R_j &= \begin{bmatrix} C^* & Z_j \end{bmatrix} \begin{bmatrix} 1 \\ Y_j h_j^* \end{bmatrix} = C^* + \mathfrak{q}_j(A^*)^{-1}\check{\mathfrak{p}}(A^*)C^* \\ &= \mathfrak{q}_j(A^*)^{-1}\Big(\mathfrak{q}_j(A^*) + \check{\mathfrak{p}}(A^*)\Big)C^* \end{aligned}$$

holds. Set $\mathfrak{p}_j = \mathfrak{q}_j + \check{\mathfrak{p}}$, which is a normalized polynomial because $\mathfrak{q}_j$ is normalized and of degree j and $\check{\mathfrak{p}}$ is of degree at most $j-1$. It remains to specify the roots of $\mathfrak{p}_j$. We transform the RAD (10.6) so that it is a (generalized) RKD with starting vector R_j and use Theorem 3.5. In order to do so, consider the matrix

$$U = \begin{bmatrix} 1 & 0 \\ Y_j h_j^* & I \end{bmatrix} \text{ with inverse } U^{-1} = \begin{bmatrix} 1 & 0 \\ -Y_j h_j^* & I \end{bmatrix}.$$

Transformation of the RAD (10.6) with U yields

$$\begin{bmatrix} C^* & Z_j \end{bmatrix} U = \begin{bmatrix} R_j & Z_j \end{bmatrix},$$

$\underline{K_j}$ is left unchanged and multiplication of U^{-1} with $\underline{H_j}$ yields

$$U^{-1}\underline{H_j} = U^{-1}\begin{bmatrix} h_j \\ \underline{H_{-j}} \end{bmatrix} = \begin{bmatrix} h_j \\ -Y_j h_j^* h_j + \underline{H_{-j}} \end{bmatrix} = \begin{bmatrix} h_j \\ \tilde{W}^* A^* Z_j \end{bmatrix},$$

where the last equality is due to Lemma 10.2. Now Theorem 3.5 implies that the roots of $\mathfrak{p}_j$ are given by $\{\lambda_1^{(j)}, \ldots, \lambda_j^{(j)}\} = \lambda(\tilde{W}^* A^* Z_j)$, which concludes the proof. □

We note that the latter result is the only one in this section for which there is no counterpart in the block case as the elements of block Krylov subspaces (3.9) correspond to matrix polynomials, which in general can not be characterized by scalar roots. For linear matrix equations, i.e. with $B = 0$, the residual formula from Theorem 10.3

simplifies considerably. It becomes

$$R_j = \frac{\prod_{i=1}^{j}(A^* + \overline{s_i}I)}{\prod_{i=1}^{j}(A^* - s_iI)}C^*$$

as $\lambda(\tilde{W}^*A^*Z_j) = \lambda(-Y_j\underline{H_{-j}}^*Y_j^{-1}) = \lambda(-\underline{H_{-j}}^*) = -\overline{s}$ holds due to Lemma 10.2. This means that the roots remain constant from step to step in the iteration, other than in the quadratic case.

We proceed with a theorem which connects the poles of the rational Krylov subspace and the eigenvalues of the projection of the matrix $A^* - X_jBB^*$. It is a generalization of parts of [52, Thm. 4.4] to oblique projections.

Theorem 10.4. *With the same assumptions as in Lemma 10.2*

$$\lambda(\tilde{W}^*(A^* - X_jBB^*)Z_j) = -\overline{s}$$

holds, i.e. the eigenvalues of the projected matrix $\tilde{W}^*(A^* - X_jBB^*)Z_j$ are the negative conjugated poles of the used Krylov subspace.

Proof. Due to Definition 3.3 and Theorem 3.4 the eigenvalues of $\underline{H_{-j}}$ are equal to the poles s and so with Lemma 10.2 and due to similarity

$$\begin{aligned}\lambda(\tilde{W}^*(A^* - X_jBB^*)Z_j) &= \lambda(-Y_j\underline{H_{-j}}^*Y_j^{-1}) \\ &= \lambda(-\underline{H_{-j}}^*) = -\overline{s}\end{aligned}$$

holds. □

11. A general projection framework

In this section we discuss how to obtain an approximate solution to the large-scale Riccati equation (1.1) via projection onto a block Krylov subspace $\mathcal{K}_j := \mathcal{K}_j^{\square}(A^*, C^*, s)$. It was shown in Section 10.4 that the Riccati ADI solution can be interpreted as the solution of a projected equation where the direction along which is projected was implicitly

defined through the residual factor. Here it is assumed that a projection direction is provided explicitly. We derive a framework to efficiently calculate the approximate solution and the norm of its residual, propose projection directions and provide an interpretation of truncated approximate solutions.

We consider the block Krylov subspace $\mathcal{K}_j$ with the orthonormal BRAD

$$A^* V_{j+1} \underline{K_j} = V_{j+1} \underline{H_j}. \tag{11.1}$$

Therein $Z_j = V_{j+1}\underline{K_j} \in \mathbb{C}^{n\times jp}$ is a basis of the Krylov subspace $\mathcal{K}_j$ and $V_{j+1} \in \mathbb{C}^{n\times(j+1)p}$ is an orthonormal basis of the augmented block Krylov subspace $\mathcal{K}_j^+ := \mathcal{K}_j^{\square+}(A^*, C^*, s)$. Let the approximate solution $X_j \approx X$ to the Riccati equation (1.1) have the representation $X_j = Z_j Y_j Z_j^*$. The goal of this section is to describe how $Y_j \in \mathbb{C}^{jp\times jp}$ can be determined via the Riccati equation (1.1) projected onto $\mathcal{K}_j$.

Let $\underline{L_j} \in \mathbb{C}^{(j+1)p\times jp}$ with regular ${\underline{L_j}}^*\underline{K_j}$ encode a subspace $\mathcal{L}_j$ of the augmented block Krylov subspace via $\mathcal{L}_j = \operatorname{span}(V_{j+1}\underline{L_j})$ with basis $W_j = V_{j+1}\underline{L_j}$. Then

$$\Pi_j = Z_j (W_j^* Z_j)^{-1} W_j^* \in \mathbb{C}^{n\times n}$$

is a projection onto $\mathcal{K}_j$ along $\mathcal{L}_j^\perp$. The small matrix Y_j is determined as the solution of the projected Riccati equation

$$\Pi_j \mathcal{R}(X_j) \Pi_j^* = 0. \tag{11.2}$$

The Riccati residual for X_j is rewritten as in Section 7.1, that is

$$\begin{aligned}\mathcal{R}(X_j) &= A^*X_j + X_jA + C^*C - X_jBB^*X_j \\ &= V_{j+1}\left(\underline{H_j}Y_j{\underline{K_j}}^* + \underline{K_j}Y_j{\underline{H_j}}^* + vv^* - \underline{K_j}Y_jS_jY_j{\underline{K_j}}^*\right)V_{j+1}^* \qquad \text{(7.3 revisited)}\end{aligned}$$

with $v \in \mathbb{C}^{(j+1)p\times p}$ such that $C^* = V_{j+1}v$ holds and $S_j = {\underline{K_j}}^* V_{j+1}^* BB^* V_{j+1}\underline{K_j}$. The basis V_{j+1} of the augmented Krylov subspace $\mathcal{K}_j^+$ is factored out in the residual formula, which allows us to restrict calculations to this $(j+1)p$-dimensional subspace: Define the

projection matrix $\pi_j \coloneqq \underline{K_j}(\underline{L_j}^*\underline{K_j})^{-1}\underline{L_j}^* \in \mathbb{C}^{(j+1)p\times(j+1)p}$ and let $u \in \mathbb{C}^{(j+1)p\times p}$ be a basis of the space $\operatorname{span}(\underline{L_j})^\perp$. Then π_j is a projection onto $\operatorname{span}(\underline{K_j})$ along $\operatorname{span}(u)$ and we find

$$\begin{aligned}\Pi_j &= V_{j+1}\underline{K_j}(\underline{L_j}^*V_{j+1}^*V_{j+1}\underline{K_j})^{-1}\underline{L_j}^*V_{j+1}^*\\ &= V_{j+1}\pi_jV_{j+1}^*\end{aligned}$$

due to the orthonormality of V_{j+1}. Herewith and using (7.3) we rewrite the projected residual for X_j as

$$\Pi_j\mathcal{R}(X_j)\Pi_j^* = V_{j+1}\pi_j\left(\underline{H_j}Y_j\underline{K_j}^* + \underline{K_j}Y_j\underline{H_j}^* + vv^* - \underline{K_j}Y_jS_jY_j\underline{K_j}^*\right)\pi_j^*V_{j+1}^*. \quad (11.3)$$

Now it is easy to see that (11.2) is equivalent to the small scale equation

$$\pi_j\left(\underline{H_j}Y_j\underline{K_j}^* + \underline{K_j}Y_j\underline{H_j}^* + vv^* - \underline{K_j}Y_jS_jY_j\underline{K_j}^*\right)\pi_j^* = 0 \quad (11.4)$$

as V_{j+1} is of full rank. To determine Y_j we intend to utilize (11.4) which mainly consists of the small matrices $\underline{K_j}$, $\underline{H_j}$ and $\underline{L_j}$.

11.1. The general projection method

We derive a Riccati equation of dimension jp which is used to solve for $Y_j \in \mathbb{C}^{jp\times jp}$. With the definition of the projection π_j one obtains from (11.4)

$$\begin{aligned}0 &= \pi_j\left(\underline{H_j}Y_j\underline{K_j}^* + \underline{K_j}Y_j\underline{H_j}^* + vv^* - \underline{K_j}Y_jS_jY_j\underline{K_j}^*\right)\pi_j^*\\ &= \underline{K_j}\Big((\underline{L_j}^*\underline{K_j})^{-1}\underline{L_j}^*\underline{H_j}Y_j + Y_j\underline{H_j}^*\underline{L_j}(\underline{K_j}^*\underline{L_j})^{-1}\\ &\quad + (\underline{L_j}^*\underline{K_j})^{-1}\underline{L_j}^*vv^*\underline{L_j}(\underline{K_j}^*\underline{L_j})^{-1} - Y_jS_jY_j\Big)\underline{K_j}^*\end{aligned}$$

Algorithm 11.1 General projection method

Input: system matrices A, B, C, set of j shifts $s \subset \mathbb{C}_+$
Output: approximate solution $X_j = V_{j+1}\underline{K_j}Y_j\underline{K_j}^*V_{j+1}^*$

1: obtain V_{j+1}, $\underline{K_j}$ and $\underline{H_j}$ from orthonormal RAD (11.1)
2: obtain v with $\overline{V_{j+1}}v = \overline{C}^*$
3: choose $\underline{L_j}$
4: solve $A_j^*Y_j + Y_jA_j + C_j^*C_j - Y_jB_jB_j^*Y_j = 0$ with A_j, B_j, and C_j as in (11.6)
5: return V_{j+1}, $\underline{K_j}$, Y_j

with the full rank matrix $\underline{K_j}$ factored out. It follows that (11.4) is equivalent to the small-scale Riccati equation

$$A_j^*Y_j + Y_jA_j + C_j^*C_j - Y_jB_jB_j^*Y_j = 0 \tag{11.5}$$

where the dense system matrices are defined as

$$\begin{aligned} A_j &:= \underline{H_j}^*\underline{L_j}(\underline{K_j}^*\underline{L_j})^{-1} \in \mathbb{C}^{jp\times jp}, \\ B_j &:= \underline{K_j}^*V_{j+1}^*B \in \mathbb{C}^{jp\times m}, \\ C_j &:= v^*\underline{L_j}(\underline{K_j}^*\underline{L_j})^{-1} \in \mathbb{C}^{p\times jp}. \end{aligned} \tag{11.6}$$

The Riccati equation (11.5) can be solved as described in Section 2.5.2. We summarize the general projection method in Algorithm 11.1.

The effectiveness of the presented projection method essentially depends on two choices: The Krylov subspace $\mathcal{K}_j^{\square}(A^*, C^*, s)$ onto which is projected (determined by the shifts s) and the space $\mathcal{L}_j^{\perp}$ along which is projected (determined by $\underline{L_j}$). For a discussion of strategies for the choice of shifts we refer the reader to [62, Sec. 5.2], [9, Sec. 4.5] and the references therein. Here we briefly investigate different choices of $\underline{L_j}$.

A quite natural choice is $\underline{L_j} = \underline{K_j}$ as in the *KSM methods, which yields an orthogonal projection, i.e. $\mathcal{L}_j = \mathcal{K}_j^{\square}(A^*, C^*, s)$. An example for an oblique projection is given by the ADI iteration as was shown in Section 10.4, though the space $\mathcal{L}_j$ is given only implicitly. It was demonstrated in Section 8 that the ADI solution only exists for Krylov subspaces with shifts $s \subset \mathbb{C}$ fulfilling $s \cap -\overline{s} = \emptyset$. In particular, if there are infinite shifts, i.e.

$C^* \in \mathcal{K}_j^{\square}(A^*, C^*, s)$, then the ADI solution does not exist.

Another choice for $\mathcal{L}_j$ is determined by $\underline{L_j} = \underline{H_j}$ which means $\mathcal{L}_j = A^*\mathcal{K}_j^{\square}(A^*, C^*, s)$ due to (11.1). This space and the orthogonal choice are two extremes of a more general family of subspaces of the augmented Krylov subspace determined by $\underline{L_j} = a\underline{H_j} - b\underline{K_j}$ with $a, b \in \mathbb{C}$ not both zero. For $p = 1$ the interpretation of this space is as follows. It consists of all vectors which can be expressed as a rational function $\mathfrak{q}(A^*)^{-1}(aA^* - b)\mathfrak{p}(A^*)C^*$ with $\mathfrak{p} \in \mathbf{\Pi}_{(j-1)}$ and with $\mathfrak{q} \in \mathbf{\Pi}_{(j)}$ having roots s. These are the rational functions in A^* multiplied with C^* with a fixed (formal) root at b/a, see [16, Prop. 3.3].

Remark 11.1. In the *KSM methods an orthogonal projection onto the (rational) Krylov subspace $\mathcal{K}_j$ with an orthonormal basis Z_j is used to obtain the approximate solution $X_j = Z_j Y_j Z_j^*$. The small matrix Y_j is obtained by imposing a Galerkin condition on the residual $\mathcal{R}(X_j)$, i.e. by solving $Z_j^*\mathcal{R}_j(X_j)Z_j = 0$, see e.g. [62] for more details. The connection to our general projection framework is as follows. Consider an orthonormal RAD (11.1) with orthonormal $\underline{K_j}$ (such a RAD can be obtained from an arbitrary RAD via QR decomposition of $\underline{K_j} = QR$ and multiplication of the RAD with R^{-1} from the right). Then $Z_j = V_{j+1}\underline{K_j}$ is an orthonormal basis of the Krylov subspace $\mathcal{K}_j$. Set $\underline{L_j} = \underline{K_j}$ so the projection is orthogonal. We then find from (11.6)

$$
\begin{aligned}
A_j &= \underline{H_j}^*\underline{K_j}(\underline{K_j}^*\underline{K_j})^{-1} = \underline{H_j}^*V_{j+1}^*V_{j+1}\underline{K_j} = \underline{K_j}^*V_{j+1}^*AV_{j+1}\underline{K_j} = Z_j^*AZ_j \\
B_j &= \underline{K_j}^*V_{j+1}^*B = Z_j^*B \\
C_j &= v^*\underline{K_j}(\underline{K_j}^*\underline{K_j})^{-1} = v^*V_{j+1}^*V_{j+1}\underline{K_j} = CZ_j
\end{aligned}
$$

due to $\underline{K_j}^*\underline{K_j} = I$ and the RAD equation (11.1). These matrices are the same as in *KSM, see [62, eq. (2.2)].

11.2. Efficient residual norm evaluation

Next we aim at making evaluation of the residual norm computationally feasible even for large n so it can be used in an iterative method as stopping criterion. We consider an unitarily invariant matrix norm and replace the residual $\mathcal{R}(X_j)$ with a small-dimensional

term having the same norm.

Let $w \in \mathbb{C}^{(j+1)p\times p}$ be a basis of $\operatorname{span}(\underline{K_j})^\perp$. Then $\overline{\pi}_j \coloneqq I - \pi_j = u(w^*u)^{-1}w^*$ is a projection onto $\operatorname{span}(u) = \operatorname{span}(\underline{L_j})^\perp$ along $\operatorname{span}(\underline{K_j})$. Note that $\mathcal{R}(X_j) = \mathcal{R}(X_j) - \Pi_j\mathcal{R}(X_j)\Pi_j^*$ holds as the projected residual is zero due to (11.2). Thus with (7.3) and (11.3) where V_{j+1} is factored out we find

$$\begin{aligned}\mathcal{R}(X_j) &= V_{j+1}\left(\underline{H_j}Y_j\underline{K_j}^* + \underline{K_j}Y_j\underline{H_j}^* + vv^* - \underline{K_j}Y_jS_jY_j\underline{K_j}^*\right)V_{j+1}^*\\ &\quad - V_{j+1}\left(\pi_j\underline{H_j}Y_j\underline{K_j}^* + \underline{K_j}Y_j\underline{H_j}^*\pi_j^* + \pi_jvv^*\pi_j^* - \underline{K_j}Y_jS_jY_j\underline{K_j}^*\right)V_{j+1}^*\\ &= V_{j+1}\left(\overline{\pi}_j\underline{H_j}Y_j\underline{K_j}^* + \underline{K_j}Y_j\underline{H_j}^*\overline{\pi}_j^* + vv^* - \pi_jvv^*\pi_j^*\right)V_{j+1}^*. \qquad (11.7)\end{aligned}$$

We observe that the quadratic term cancels out. For the constant term it holds

$$\begin{aligned}vv^* - \pi_jvv^*\pi_j^* &= 0.5\cdot(vv^* + vv^*\pi_j^* - \pi_jvv^* - \pi_jvv^*\pi_j^*)\\ &\quad + 0.5\cdot(vv^* - vv^*\pi_j^* + \pi_jvv^* - \pi_jvv^*\pi_j^*)\\ &= \overline{\pi}_jvv^*(I - 0.5\cdot\overline{\pi}_j)^* + (I - 0.5\cdot\overline{\pi}_j)vv^*\overline{\pi}_j^*\end{aligned}$$

because of

$$\begin{aligned}0.5\cdot(vv^* + vv^*\pi_j^* - \pi_jvv^* - \pi_jvv^*\pi_j^*) &= 0.5\cdot(I-\pi_j)vv^*(I+\pi_j)^*\\ &= \overline{\pi}_jvv^*\cdot 0.5\cdot(I + I - \overline{\pi}_j)^*\\ &= \overline{\pi}_jvv^*(I - 0.5\cdot\overline{\pi}_j)^*.\end{aligned}$$

Herewith (11.7) can be further manipulated to obtain

$$\begin{aligned}\mathcal{R}(X_j) &= V_{j+1}\big(\overline{\pi}_j(\underline{H_j}Y_j\underline{K_j}^* + vv^*(I - 0.5\cdot\overline{\pi}_j)^*)\\ &\quad + (\underline{K_j}Y_j\underline{H_j}^* + (I - 0.5\cdot\overline{\pi}_j)vv^*)\overline{\pi}_j^*\big)V_{j+1}^*\\ &= V_{j+1}\left(ux^* + xu^*\right)V_{j+1}^* \qquad (11.8)\end{aligned}$$

Algorithm 11.2 Residual norm calculation in general projection method

Input: $\underline{K_j}$, $\underline{H_j}$, $\underline{L_j}$, solution Y_j of (11.5) and v with $V_{j+1}v = C^*$

Output: residual norm $\|\mathcal{R}(X_j)\|$

1: compute basis u of $\operatorname{span}(\underline{L_j})^\perp$

2: compute basis w of $\operatorname{span}(\underline{K_j})^\perp$

3: compute $x = \underline{K_j} Y_j \underline{H_j}^* w(u^*w)^{-1} + (v - 0.5 \cdot u(w^*u)^{-1}w^*v)v^*w(u^*w)^{-1}$

4: compute economy-size QR decomposition $QR = [u, x]$

5: return $\left\| R \begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix} R^* \right\|$

due to $\overline{\pi}_j = u(w^*u)^{-1}w^*$ and with

$$\begin{aligned} x &\coloneqq \underline{K_j} Y_j \underline{H_j}^* w(u^*w)^{-1} + (I - 0.5 \cdot \overline{\pi}_j)vv^*w(u^*w)^{-1} \\ &= \underline{K_j} Y_j \underline{H_j}^* w(u^*w)^{-1} + (v - 0.5 \cdot u(w^*u)^{-1}w^*v)v^*w(u^*w)^{-1}. \end{aligned} \tag{11.9}$$

The residual formulation (11.8) is a generalization of the expression in [62, Prop. 5.3]. We have factored out the orthonormal basis V_{j+1} of the augmented Krylov subspace, which allows calculation of the residual norm using only the low-dimensional inner quantities $u, x \in \mathbb{C}^{(j+1)p \times p}$. Let $QR = [u, x]$ be an economy-size QR decomposition. Then

$$\begin{aligned} \|\mathcal{R}(X_j)\| &= \|V_{j+1}(ux^* + xu^*)V_{j+1}^*\| \\ &= \left\| V_{j+1} QR \begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix} R^* Q^* V_{j+1}^* \right\| \\ &= \left\| R \begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix} R^* \right\| \end{aligned} \tag{11.10}$$

holds for any unitarily invariant norm as V_{j+1} and Q are orthonormal.

Remark 11.2. In the literature usually Krylov subspaces with $C^* \in \mathcal{K}_j(A^*, C^*, s)$ (i.e. with a shift $\infty \in s$) are used, see e.g. [62] and the references therein. This means that there exists a block vector $\tilde{v}$ with $C^* = V_{j+1}\underline{K_j}\tilde{v}$, so $v = \underline{K_j}\tilde{v} \in \operatorname{span}(\underline{K_j})$ holds. In this case (11.9) reduces to $x = \underline{K_j} Y_j \underline{H_j}^* w(u^*w)^{-1}$ because the constant term cancels out in (11.7) as π_j is a projection onto $\operatorname{span}(\underline{K_j})$ and so $\pi_j v = v$ holds.

Algorithm 11.3 General projection method (iterative)

Input: system matrices A, B, C, set of shifts $s \subset \mathbb{C}_+$
Output: approximate solution $X_j = V_{j+1}\underline{K_j}Y_j\underline{K_j}^*V_{j+1}^*$

initialize $V_1 = \operatorname{orth}(C^*)$, $\hat{B}_0 = V_1^*B$, $j = 1$
compute $\alpha \in \mathbb{C}^{p\times p}$ with $V_1\alpha = C^*$
while not converged **do**
  obtain new shift μ from s
  expand orthonormal BRAD (11.1), obtain $V_{j+1} = [V_j,\ \hat{Z}_j]$, $\underline{K_j}$ and $\underline{H_j}$
  choose $\underline{L_j}$, set $l_j = [I_p, 0, \ldots, 0]\underline{L_j}$
  compute $A_j = \underline{H_j}^*\underline{L_j}(\underline{K_j}^*\underline{L_j})^{-1}$
  compute $\hat{B}_j = \begin{bmatrix}\hat{B}_{j-1}\\ \hat{Z}_j^*B\end{bmatrix}$
  compute $B_j = \underline{K_j}^*\hat{B}_j$
  compute $C_j = \alpha^* l_j(\underline{K_j}^*\underline{L_j})^{-1}$
  solve $A_j^*Y_j + Y_jA_j + C_j^*C_j - Y_jB_jB_j^*Y_j = 0$
  compute $\|\mathcal{R}(X_j)\|$ via Algorithm 11.2
  $j = j + 1$
end while
return V_{j+1}, $\underline{K_j}$, Y_j

Remark 11.3. From (11.8) we find that the rank of the residual is at most $2p$. There are only two scenarios in which the residual rank is smaller. On the one hand the residual can be of rank p, which happens for the ADI solution as shown in Section 10.4. On the other hand, a rank-0 residual is obtained for an exact solution. The rank-$2p$ property has also been investigated in [68, Sec. 5.2] for Lyapunov equations. It was called *dilemma of RKSM* as in general the norm minimizing approximate solution yields a residual with larger rank.

The calculation of the residual norm as described above is summarized in Algorithm 11.2. It enables us to calculate the approximate solution iteratively as stated in Algorithm 11.3 where the iteration is stopped when the norm of the residual is sufficiently small. As C^* is the starting vector of the Krylov subspace we can express it as a linear combination of the first p columns of V_{j+1}. Thus it holds $C^* = V_1\alpha$ for $\alpha = [I_p, 0, \ldots, 0]v \in \mathbb{C}^{p\times p}$. Further, to speed up calculation of $B_j = \underline{K_j}^*V_{j+1}^*B$ we store $\hat{B}_j = V_{j+1}^*B$ which can be updated cheaply when the BRAD is expanded.

11.3. Truncation of the approximate solution

The general projection method generates an approximate solution X_j which may be indefinite due to rounding errors and can be approximated by a matrix of lower rank. We consider an eigendecomposition of the symmetric matrix X_j and truncate the eigenvectors corresponding to negative and small eigenvalues to obtain the truncated approximate solution $\tilde{X}_j$, which is positive semidefinite and of lower rank. The key question is of course where to truncate the eigenvalue decomposition. We propose to use the norm of the Riccati residual of the truncated approximate solution as an indicator. For an efficient evaluation of the residual norm we provide an interpretation of the truncated approximate solution as the solution of the Riccati equation projected to a subspace $\tilde{\mathcal{K}}_j \subseteq \mathcal{K}_j$ of dimension $r \leq jp$ which is determined by the decomposition of X_j.

Let $A^*V_{j+1}\underline{K_j} = V_{j+1}\underline{H_j}$ be an orthonormal BRAD as in (11.1) and let $Y_j \in \mathbb{C}^{jp\times jp}$ be the solution of (11.4), so $X_j = V_{j+1}\underline{K_j}Y_j\underline{K_j}^*V_{j+1}^*$ is the solution of the projected Riccati equation (11.2). Consider the decomposition

$$\underline{K_j}Y_j\underline{K_j}^* = \begin{bmatrix}\underline{\tilde{K}_j} & \underline{\hat{K}_j}\end{bmatrix}\begin{bmatrix}\tilde{Y}_j & 0\\ 0 & \hat{Y}_j\end{bmatrix}\begin{bmatrix}\underline{\tilde{K}_j}^*\\ \underline{\hat{K}_j}^*\end{bmatrix} \tag{11.11}$$

with $\tilde{Y}_j \in \mathbb{C}^{r\times r}$, $\underline{\tilde{K}_j} \in \mathbb{C}^{(j+1)p\times r}$, $\underline{\hat{K}_j} \in \mathbb{C}^{(j+1)p\times(jp-r)}$ and $\operatorname{span}(\underline{K_j}) = \operatorname{span}\Big(\Big[\underline{\tilde{K}_j}, \underline{\hat{K}_j}\Big]\Big)$. Define the truncated approximate solution

$$\tilde{X}_j \coloneqq V_{j+1}\underline{\tilde{K}_j}\tilde{Y}_j\underline{\tilde{K}_j}^*V_{j+1}^*.$$

In the following we do not restrict ourselves to eigendecompositions but use the more general decomposition (11.11). Of course we have an eigendecomposition in mind. We suppose that the part that is truncated is determined by $\hat{Y}_j$, i.e. the negative and small eigenvalues are contained in $\hat{Y}_j$. Note that in (11.11) already p eigenvectors corresponding to the eigenvalue zero are truncated as $\underline{K_j} \in \mathbb{C}^{(j+1)p\times jp}$ is a rectangular matrix.

Next we discuss the connection of the truncated approximate solution $\tilde{X}_j$ and pro-

jection onto the subspace $\tilde{\mathcal{K}}_j \coloneqq \operatorname{span}(V_{j+1}\underline{\tilde{K}_j})$. Set $\tilde{Z}_j \coloneqq V_{j+1}\underline{\tilde{K}_j}$ and $\tilde{W}_j \coloneqq V_{j+1}\underline{\tilde{L}_j}$ where $\underline{\tilde{L}_j}$ is a basis of the space $\operatorname{span}(\operatorname{span}(\underline{L_j})^\perp + \operatorname{span}(\underline{\hat{K}_j}))^\perp$. Then

$$\begin{aligned}\tilde{\Pi}_j &\coloneqq \tilde{Z}_j(\tilde{W}_j^*\tilde{Z}_j)^{-1}\tilde{W}_j^* \\ &= V_{j+1}\underline{\tilde{K}_j}(\underline{\tilde{L}_j}^*\underline{\tilde{K}_j})^{-1}\underline{\tilde{L}_j}^* V_{j+1}^* \\ &= V_{j+1}\tilde{\pi}_j V_{j+1}^*\end{aligned}$$

with $\tilde{\pi}_j \coloneqq \underline{\tilde{K}_j}(\underline{\tilde{L}_j}^*\underline{\tilde{K}_j})^{-1}\underline{\tilde{L}_j}^*$ is an (in general oblique) projection onto $\tilde{\mathcal{K}}_j$. By construction $\tilde{\pi}_j = \tilde{\pi}_j\pi_j$ is satisfied. Let $T_1 \in \mathbb{C}^{jp\times r}$ be such that $\underline{K_j}T_1 = \underline{\tilde{K}_j}$ holds. With the pseudoinverse $\underline{K_j}^+$ we thus have $T_1 = \underline{K_j}^+\underline{\tilde{K}_j}$. Define $\underline{\tilde{H}_j} \coloneqq \underline{H_j}T_1$, which gives the RAD-like relation

$$\begin{aligned}A^*V_{j+1}\underline{\tilde{K}_j} &= A^*V_{j+1}\underline{K_j}T_1 \\ &= V_{j+1}\underline{H_j}T_1 \\ &= V_{j+1}\underline{\tilde{H}_j}.\end{aligned}$$

We now state the main result of this subsection.

Theorem 11.4. *The truncated approximate solution $\tilde{X}_j = V_{j+1}\underline{\tilde{K}_j}\tilde{Y}_j\underline{\tilde{K}_j}^*V_{j+1}^*$ satisfies the projected equation*

$$\tilde{\Pi}_j\mathcal{R}_j(\tilde{X}_j)\tilde{\Pi}_j^* = 0. \tag{11.12}$$

That is, for $\tilde{Y}_j \in \mathbb{C}^{r\times r}$ the equation

$$\tilde{\pi}_j\left(\underline{\tilde{H}_j}\tilde{Y}_j\underline{\tilde{K}_j}^* + \underline{\tilde{K}_j}\tilde{Y}_j\underline{\tilde{H}_j}^* + vv^* - \underline{\tilde{K}_j}\tilde{Y}_j\tilde{S}_j\tilde{Y}_j\underline{\tilde{K}_j}^*\right)\tilde{\pi}_j^* = 0 \tag{11.13}$$

holds, where $\tilde{S}_j \coloneqq \underline{\tilde{K}_j}^*V_{j+1}^*BB^*V_{j+1}\underline{\tilde{K}_j}$.

Proof. The equivalence of (11.12) and (11.13) follows in full analogy to the equivalence of (11.2) and (11.4) by using the corresponding quantities with a tilde. We apply the

projection $\tilde{\pi}_j$ with $\tilde{\pi}_j \pi_j = \tilde{\pi}_j$ to (11.4) which gives us

$$\tilde{\pi}_j \left(\underline{H_j} Y_j \underline{K_j}^* + \underline{K_j} Y_j \underline{H_j}^* + vv^* - \underline{K_j} Y_j S_j Y_j \underline{K_j}^* \right) \tilde{\pi}_j^* = 0$$

To show equivalence of this equation and (11.13) each of the terms is investigated separately. First note that

$$\begin{aligned} \tilde{\pi}_j \underline{K_j} Y_j \underline{K_j}^* &= \tilde{\pi}_j \begin{bmatrix} \underline{\tilde{K}_j} & \underline{\hat{K}_j} \end{bmatrix} \begin{bmatrix} \tilde{Y}_j & 0 \\ 0 & \hat{Y}_j \end{bmatrix} \begin{bmatrix} \underline{\tilde{K}_j}^* \\ \underline{\hat{K}_j}^* \end{bmatrix} \\ &= \begin{bmatrix} \underline{\tilde{K}_j} & 0 \end{bmatrix} \begin{bmatrix} \tilde{Y}_j & 0 \\ 0 & \hat{Y}_j \end{bmatrix} \begin{bmatrix} \underline{\tilde{K}_j}^* \\ \underline{\hat{K}_j} \end{bmatrix} \\ &= \underline{\tilde{K}_j} \tilde{Y}_j \underline{\tilde{K}_j}^* \\ &= \tilde{\pi}_j \underline{\tilde{K}_j} \tilde{Y}_j \underline{\tilde{K}_j}^* \end{aligned} \tag{11.14}$$

holds due to (11.11), $\tilde{\pi}_j \underline{\tilde{K}_j} = \underline{\tilde{K}_j}$ and $\tilde{\pi}_j \underline{\hat{K}_j} = 0$. For the first term we find

$$\begin{aligned} \tilde{\pi}_j \underline{H_j} Y_j \underline{K_j}^* \tilde{\pi}_j &= \tilde{\pi}_j \underline{H_j K_j}^+ \underline{K_j} Y_j \underline{K_j}^* \tilde{\pi}_j^* \\ &= \tilde{\pi}_j \underline{H_j K_j}^+ \underline{\tilde{K}_j} \tilde{Y}_j \underline{\tilde{K}_j}^* \tilde{\pi}_j \\ &= \tilde{\pi}_j \underline{H_j} T_1 \tilde{Y}_j \underline{\tilde{K}_j}^* \tilde{\pi}_j^* \\ &= \tilde{\pi}_j \underline{\tilde{H}_j} \tilde{Y}_j \underline{\tilde{K}_j}^* \tilde{\pi}_j^* \end{aligned}$$

with the transposed of (11.14), $\underline{K_j}^+ \underline{K_j} = I$ and $\underline{\tilde{H}_j} = \underline{H_j} T_1$. The second term is the transposed of the first one. The constant term is left unchanged. For the quadratic term we have

$$\begin{aligned} \tilde{\pi}_j \underline{K_j} Y_j S_j Y_j \underline{K_j}^* \tilde{\pi}_j^* &= \tilde{\pi}_j \underline{K_j} Y_j \underline{K_j}^* V_{j+1}^* B B^* V_{j+1} \underline{K_j} Y_j \underline{K_j}^* \tilde{\pi}_j^* \\ &= \tilde{\pi}_j \underline{\tilde{K}_j} \tilde{Y}_j \underline{\tilde{K}_j}^* V_{j+1}^* B B^* V_{j+1} \underline{\tilde{K}_j} \tilde{Y}_j \underline{\tilde{K}_j}^* \tilde{\pi}_j^* \\ &= \tilde{\pi}_j \underline{\tilde{K}_j} \tilde{Y}_j \tilde{S}_j \tilde{Y}_j \underline{\tilde{K}_j}^* \tilde{\pi}_j^* \end{aligned}$$

due to (11.14). Putting these terms together we see that (11.13) holds, which completes

the proof. □

The preceding theorem allows us to efficiently evaluate the norm of the Riccati residual for the truncated approximate solution $\tilde{X}_j$ in the same way as described in Section 11.2 by using $\underline{\tilde{K}_j}$, $\underline{\tilde{H}_j}$, $\underline{\tilde{L}_j}$ and $\tilde{Y}_j$ instead of the values without tilde. The solution $\tilde{Y}_j$ of (11.13) is given by the decomposition (11.11), so no additional small-scale Riccati equation has to be solved. The matrices $\underline{\tilde{L}_j}$ and $\underline{\tilde{K}_j}$ consist of r columns, thus we have $u, w \in \mathbb{C}^{(j+1)p \times (j+1)p-r}$ in Algorithm 11.2. Hence a smaller dimension r of $\tilde{\mathcal{K}}_j$ means more columns in u and w and the rank of the residual matrix becomes larger than $2p$. The efficient residual norm evaluation allows us to use an eigenvalue decomposition of $\underline{K_j} Y_j \underline{K_j}$ and increase the number of truncated eigenvalues iteratively as long as the residual norm is sufficiently small.

11.4. Generalized Riccati equations

For the generalized Riccati equation

$$A^*XE + E^*XA + C^*C - E^*XBB^*XE = 0 \qquad \text{(10.1 revisited)}$$

we have to apply modifications similar to those in Section 10.1. Again, essentially the equivalent equation

$$E^{-*}A^*X + XAE^{-1} + E^{-*}C^*CE^{-1} - XBB^*X = 0 \qquad \text{(10.2 revisited)}$$

is solved. The orthonormal RAD used in Algorithm 11.1 and Algorithm 11.3 thus becomes

$$\begin{aligned} & E^{-*}A^*V_{j+1}\underline{K_j} = V_{j+1}\underline{H_j} \\ \Leftrightarrow \qquad & A^*V_{j+1}\underline{K_j} = E^*V_{j+1}\underline{H_j} \end{aligned}$$

with starting vector $E^{-*}C^*$. Note that the residual norm calculation becomes more sophisticated because with Algorithm 11.2 the residual of (10.2) is considered. It can be

transformed into the residual of (10.1) via multiplication with E^* from the left and E from the right (as this is how (10.2) is transformed into (10.1)). Herewith (11.8) becomes

$$E^*V_{j+1}\left(ux^* + xu^*\right)V_{j+1}^*E. \tag{11.15}$$

To calculate the norm of (11.15) efficiently, let $QR = E^*V_{j+1}[u, x]$ be an economy-size QR decomposition. Then

$$\begin{aligned}\|E^*V_{j+1}(ux^* + xu^*)V_{j+1}^*E\| &= \left\|QR\begin{bmatrix}0 & I_p\\ I_p & 0\end{bmatrix}R^*Q^*\right\| \\ &= \left\|R\begin{bmatrix}0 & I_p\\ I_p & 0\end{bmatrix}R^*\right\|\end{aligned}$$

is the residual norm of (10.1) (cf. (11.10)).

12. Numerical experiments

We perform numerical experiments to compare the Riccati RAD iteration as in Algorithm 9.1 with the Lyapunov RADI iteration as in Algorithm 9.2, show the effects of our parallel approach and compare the ADI-type algorithms with the general projection method (GPM). The ADI-type algorithms were implemented including the modifications described in Section 10.1 to handle generalized Riccati equations (10.1) with an additional system matrix E, with realification and the possibility to use parallelization as described in Section 9.3.

Note that the Lyapunov RADI iteration represents the RADI iteration, as it is just a more general variant of the RADI iteration from [9], see also Remark 9.2. In a comparison of the approximate solution of our implementation and of the RADI implementation from M-M.E.S.S.-2.0 [53] we only found differences in the order of the machine precision, but our implementation was more efficient in terms of computational time. We therefore choose to compare only the timings for the R^{2}ADi and the Lyapunov RADI iteration,

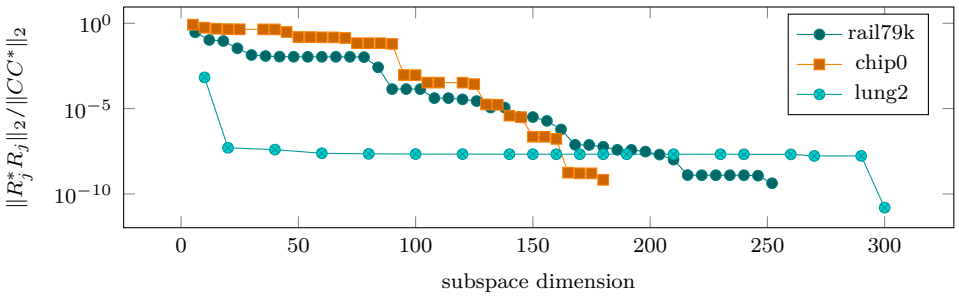

Figure 12.1.: Relative residual norms for the ADI iterates with shifts generated by M-M.E.S.S.

but not for the RADI iteration. This allows a fair performance comparison, as the implementations of our methods share large parts of their code.

The convergence of the iteration considerably depends on the choice of shifts which determine the Krylov subspace used for the approximate solution. We used the M-M.E.S.S.-2.0 implementation of the RADI iteration with the shift strategy *residual Hamiltonian shifts* as described in Section 10.3 (denoted *gen-ham-opti* in M-M.E.S.S.-2.0) and parameter $l = 6p$ to precompute the shifts which were then used in our numerical experiments. The RADI iteration was stopped when the relative residual norm $\|R_j R_j^*\|_2 / \|C^* C\|_2 = \|R_j^* R_j\|_2 / \|CC^*\|_2$ became smaller than 10^{-9}. Due to the precomputed shifts the Riccati ADI solution is fixed. This allows us to employ our parallelization approach and compare the accuracy of approximate solutions obtained with different numbers of parallel threads. However, as shifts are precomputed, no timings for the shift calculation are presented, although this may contribute considerably to the iteration time. For a discussion of the effects of different shift strategies and different Riccati ADI methods we refer to the numerical experiments in [9, Sec. 5].

We used the three examples rail79k with $n = 79841$, chip0 with $n = 20082$ and lung2 with $n = 109460$ as introduced in Section 2.6. All these examples are real valued, so all iteration steps were executed with realification in case of complex shifts.

12.1. Comparison of R²ADi and Lyapunov RADI iteration

All numerical experiments in this subsection were executed using MATLAB 2020b on an Intel® Core™ i7-5600U CPU @ 2.60 GHz with 12 GB RAM.

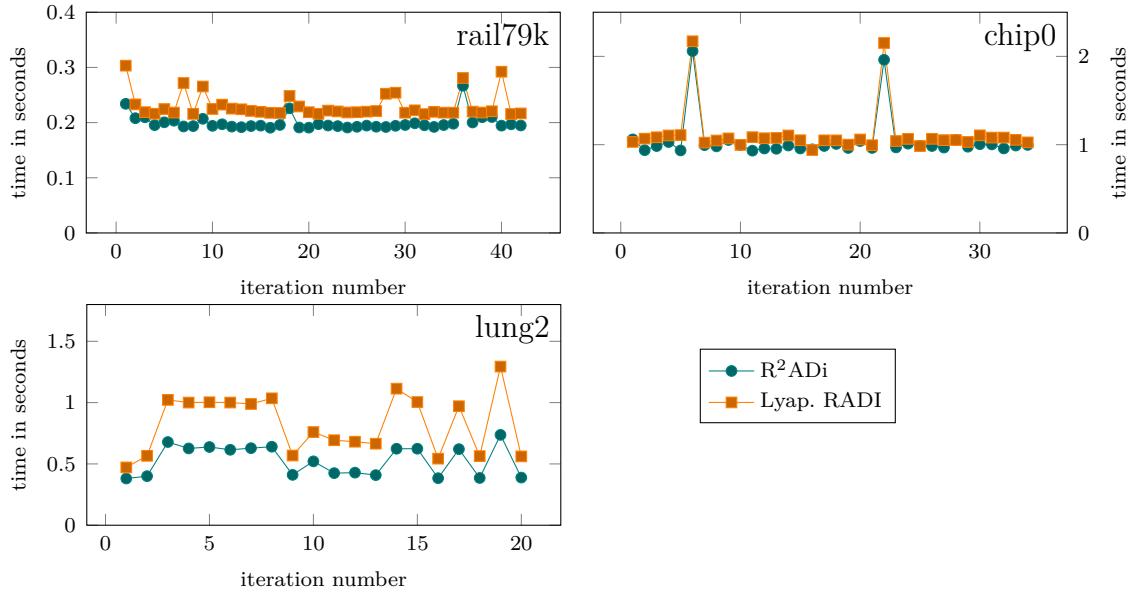

Figure 12.2.: Computational time for the RAD expansion in each iteration step of the R^{2}ADi and Lyapunov RADI iteration.

In Figure 12.1 the convergence behavior for the three examples is plotted. Table 12.1 contains the computational time for the different parts of the iterations. The expansion of the RADs is the most expensive part due to the necessary solves of linear systems. The time needed for the RAD expansion in every step of the iterations is shown in Figure 12.2, where the spikes come from the more expensive solves when complex shifts are involved. We observe that the Lyapunov RADI iteration needs more time for the linear solves. This is due to the SMW formula, where systems with $m + p$ right-hand sides have to be solved instead of p right-hand sides in the R^{2}ADi. For instance in the lung2 example with $m = 10$ the additional costs for the SMW formula make system solves 60 % more expensive than in the R^{2}ADi, while in the chip0 example with $m = 1$

	subsp.	R^{2}ADi			Lyap. RADI		
	dim.	RAD exp.	misc.	total	RAD exp.	misc.	total
rail79k	252	8.4	0.9	**9.3**	9.7	0.6	**10.3**
chip0	180	35.6	0.1	**35.7**	38.0	0.1	**38.1**
lung2	300	10.6	0.6	**11.2**	16.5	0.4	**16.9**

Table 12.1.: Times in seconds for different parts of the ADI iterations.

the additional costs are only small.

All other parts of the iterations are aggregated under *misc*. The slightly larger times for the R^{2}ADi are due to the multiplication with the Krylov basis Z_{j+1} in the residual update $R_{j+1} = C^* + Z_{j+1}Y_{j+1}h^*_{j+1}$ in line 8 of Algorithm 9.1. However, in all examples considered here the cheaper system solves and the not necessary update of K_j in the R^{2}ADi compensate for this additional costs.

12.2. Effect of parallelization in R^{2}ADi

For the experiments in this subsection we used MATLAB 2018b on four Intel® Xeon® CPU E7-4880 v2 @ 2.50 GHz with altogether 60 CPU cores and 1 TB RAM. For the parallel expansion of the RADs the *parfor* command in MATLAB was utilized. All calculations were performed with the Riccati RAD iteration Algorithm 9.1.

To parallelize the system solves, multiple shifts have to be available. This is the case here as all shifts were precomputed, which allows us to compare the performance and accuracy of the parallel with the serial approach. In practice the shifts are calculated one after another during the iteration as described in Section 10.3. An efficient shift strategy which obtains multiple shifts per iteration step has yet to be found.

In Figure 12.3 the computational times for the iteration and speedup factors are plotted against the number of parallel system solves in the RAD expansion step. The speedup factor is the iteration time for an iteration without parallelization and a *for* loop divided by the iteration time needed with parallel system solves and MATLABs

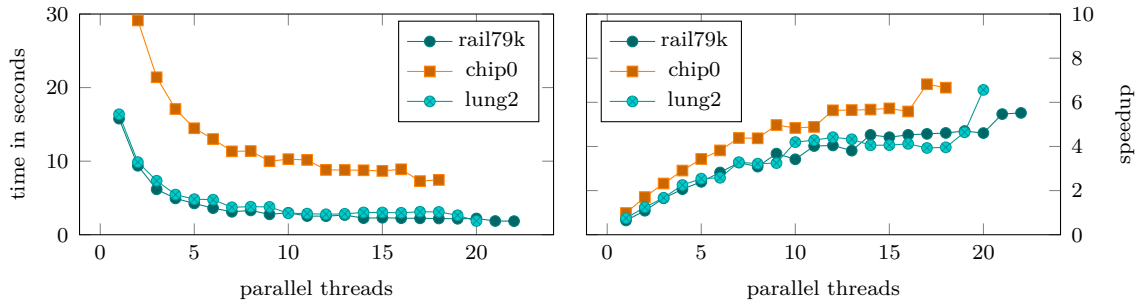

Figure 12.3.: Times and speedups for R^{2}ADi with parallel RAD expansion.

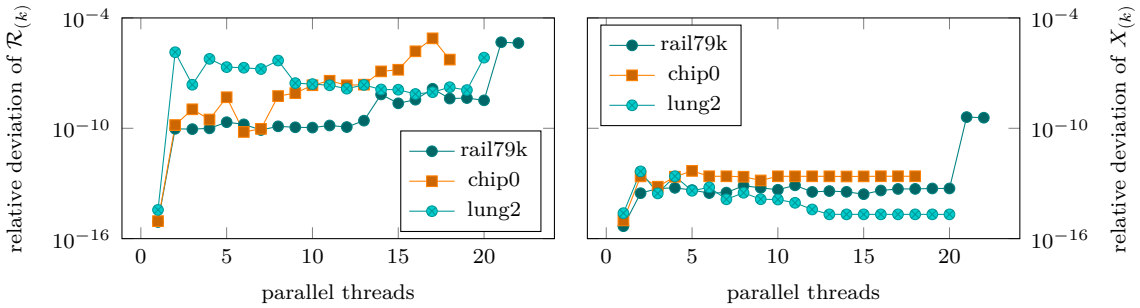

Figure 12.4.: Relative deviation of parallel residuals and approximants from serial results.

parfor loop. Due to the overhead introduced by the *parfor* command the serial iteration with a *for* loop is faster than the same serial iteration with the *parfor* loop. Thus the speedup factor for one thread is smaller than one. Further, only the system solves are executed in parallel but not the other variable updates. As they take up to about 10 % of the calculation time in the rail79k and lung2 examples the possible maximal speedup is limited. Indeed we observe a moderate speedup for the rail79k and lung2 examples, while for the chip0 example the speedup is higher.

Besides the performance gain we also investigate the accuracy of the parallel calculations. We compare the parallel iterates with the serial iterates. Let a subscript (k) denote the number of parallel threads utilized to obtain the variable. We calculated the relative deviation of the residuals obtained with k parallel threads from the residual obtained with the serial iteration, i.e.

$$\frac{\|\mathcal{R}_{(1)} - \mathcal{R}_{(k)}\|_2}{\|\mathcal{R}_{(1)}\|_2}.$$

The second quantity we use to indicate the accuracy is the relative deviation of the parallel approximants $X_{(k)}$ from the serial approximant $X_{(1)}$

$$\frac{\|X_{(1)} - X_{(k)}\|_2}{\|X_{(1)}\|_2}.$$

A direct calculation of the norms of the involved matrices is infeasible due to the large dimensions. We thus exploit the factorized form of the residual and the approximants,

i.e. $\mathcal{R}_{(k)} = R_{(k)}R_{(k)}^*$ and $X_{(k)} = \hat{Z}_{(k)}\hat{Z}_{(k)}^*$ as in Remark 9.1. Let $QS = [\hat{Z}_{(1)}, \hat{Z}_{(k)}]$ be an economy-size QR decomposition with upper triangular S. Then due to the unitary invariance of the norm it holds

$$\|X_{(1)} - X_{(k)}\|_2 = \left\| S \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix} S^* \right\|_2$$

and only the norm of a small matrix must be computed. For the denominator we use the formula $\|\hat{Z}_{(k)}\hat{Z}_{(k)}^*\|_2 = \|\hat{Z}_{(k)}^*\hat{Z}_{(k)}\|_2$. We proceed in the same way for the residual deviation.

The results are displayed in Figure 12.4. We observe that the relative deviations from the residual are below 10^{-5} for all examples. The relative deviation of the final approximation is even smaller than 10^{-12} for all examples and fewer than twenty parallel threads. All in all the effects of parallelization to the accuracy of the results appear to be very little. Thus parallel system solves are a reasonable technique to speed up computations.

12.3. Comparison of GPM and ADI

For the experiments in this subsection we used MATLAB 2020b on an Intel® Core™ i7-5600U CPU @ 2.60 GHz with 12 GB RAM. The same approximation space as in the previous subsections is used, i.e. the Krylov subspace determined by the shifts generated with the RADI iteration from M-M.E.S.S. We compare the convergence and timings of the Riccati RAD iteration with the general projection method Algorithm 11.3. The behavior of the GPM iterates for $\underline{L_j} = a\underline{H_j} - b\underline{K_j}$ and different choices of a and b is investigated. See also [29] for a comparison of the ADI iteration and orthogonal projection to the same approximation space for Lyapunov equations.

The orthonormal BRAD for the GPM is generated using the function *rat_krylov* from the *Rational Krylov Toolbox* [17] in version 2.9. It turns out that an iterative calculation (i.e. expansion of the RAD in every step) is considerably slower than directly calculating the whole RAD. Therefore the orthogonal RAD was generated prior to the start of

the iteration, which is possible here as the shifts are precomputed. Then increasing parts of the RAD were used during the iteration. In practice one might want to use a more efficient implementation of the RAD expansion. The *rat_krylov* function further supports generalized Riccati equations with an additional system matrix E, block vectors $C^* \in \mathbb{C}^{n\times p}$ and realification. All occurring small-scale Riccati equations are solved with MATLABs *icare*. To compute the Riccati ADI solution we use the R^{2}ADi Algorithm 9.1 which needs no orthonormal basis but exploits the rank-p residual condition.

The execution times for the different parts of GPM are displayed in Table 12.2. Note that the choice of $\underline{L_j}$ does not affect the timings. For all three examples GPM needs more time than R^{2}ADi. While the run time difference is small for the chip0 example we see that for the rail79k and lung2 GPM runs three to four times as long as R^{2}ADi. This is due to the following reasons. To generate the RAD, in both algorithms the same linear systems of equations are solved, but in GPM additionally orthonormalization of the basis is necessary. In R^{2}ADi the Lyapunov equation (9.5) of constant, small size has to be solved in every step, while in GPM the solution of the small-scale Riccati equation (11.5) of growing dimension has to be calculated.

For the two examples with an additional system matrix E the necessary time to calculate the residual norm of (10.1) is stated in Table 12.2. The residual norm of (10.2) can be obtained faster: Only 0.3 seconds are needed for the rail79k example and only 0.1 seconds for the chip0 example. This is due to the multiplication $E^*V_{j+1}[u, x]$ which is necessary in the former case, but not in the latter, see Section 11.4.

The convergence plots in Figure 12.5 are displayed for the residual norm of (10.1) on the left and of (10.2) on the right. We stress again that the shifts were chosen such

	subsp.	GPM					R^{2}ADi
	dim.	orth. RAD	small Riccati	res.	misc.	total	total
rail79k	252	26.8	11.1	4.0	0.7	**42.6**	**9.3**
chip0	180	37.0	4.9	0.5	0.1	**42.5**	**35.7**
lung2	300	31.4	6.5	0.2	0.7	**38.8**	**11.2**

Table 12.2.: Times in seconds for different parts of the GPM iteration and total time for R^{2}ADi.

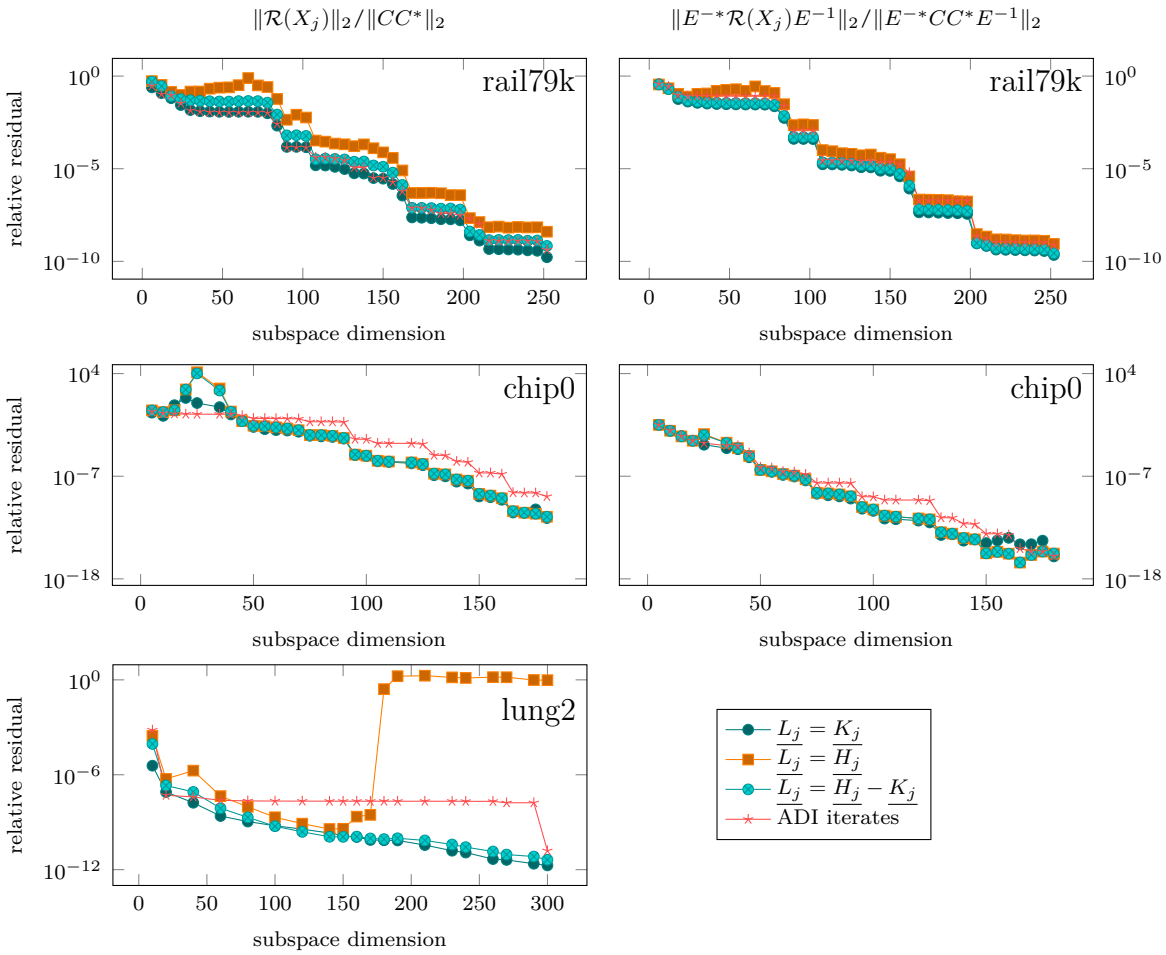

Figure 12.5.: Residual plots of GPM iteration for different choices of $\underline{L_j}$ and for the ADI iteration.

that the relative residual norm of (10.1) for the ADI iterates is below 10^{-9} and that the same shifts were used in GPM. We observe that the convergence behavior on the left and on the right is quite similar for the rail79k example but varies for the chip0 example. The four investigated iterations show a similar performance in the rail79k and chip0 examples. In the lung2 example the GPM iterations with $\underline{L_j} = \underline{K_j}$ and with $\underline{L_j} = \underline{H_j} - \underline{K_j}$ converge regularly and the one with $\underline{L_j} = \underline{H_j}$ diverges. The ADI iterates mostly stagnate until finally the desired accuracy is reached. In all iterations the final residual is smallest for the orthogonal projection. A drawback of the GPM is that sometimes the positive definiteness of the solution of the small-scale Riccati equation (11.5) is lost numerically. For the rail79k example positive definiteness was lost after

reaching a subspace dimension of 246 (for $\underline{L_j} = \underline{K_j}$) respectively of 240 (for $\underline{L_j} = \underline{H_j}$ and $\underline{L_j} = \underline{H_j} - \underline{K_j}$). Positive definiteness in the chip0 example was lost after a subspace dimension of 120, 115 and 110 was reached (for $\underline{L_j} = \underline{K_j}$, $\underline{L_j} = \underline{H_j}$ and $\underline{L_j} = \underline{H_j} - \underline{K_j}$). In the lung2 example Y_j was positive definite during all steps.

12.4. Truncated approximate GPM solution

Next we use the technique described in Section 11.3 to calculate the norm of the truncated approximate GPM solution. For this the approximate solutions for the orthogonal projections (i.e. with $\underline{L_j} = \underline{K_j}$) from the previous subsection are postprocessed. Consider an eigenvalue decomposition of the symmetric matrix

$$\underline{K_j} Y_j \underline{K_j}^* = \begin{bmatrix} q_1 & \cdots & q_{jp} \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_{jp} \end{bmatrix} \begin{bmatrix} q_1^* \\ \vdots \\ q_{jp}^* \end{bmatrix} \in \mathbb{C}^{(j+1)p \times (j+1)p}$$

where the q_i are orthonormal, the eigenvalues are ordered decreasingly, i.e. $\lambda_1 \geq \cdots \geq \lambda_{jp}$, and where the p eigenvectors corresponding to the zero eigenvalues are already truncated. The truncated approximate solution is given by $\tilde{X}_j = V_{j+1} \underline{\tilde{K}_j} \tilde{Y}_j \underline{\tilde{K}_j}^* V_{j+1}^*$ with

$$\underline{\tilde{K}_j} = \begin{bmatrix} q_1 & \cdots & q_r \end{bmatrix} \text{ and } \tilde{Y}_j = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_r \end{bmatrix}.$$

To obtain the decomposition (11.11) we further set

$$\underline{\hat{K}_j} = \begin{bmatrix} q_{r+1} & \cdots & q_{jp} \end{bmatrix} \text{ and } \hat{Y}_j = \begin{bmatrix} \lambda_{r+1} & & \\ & \ddots & \\ & & \lambda_{jp} \end{bmatrix}.$$

In Figure 12.6 the residual norm of the GPM method from the previous subsection

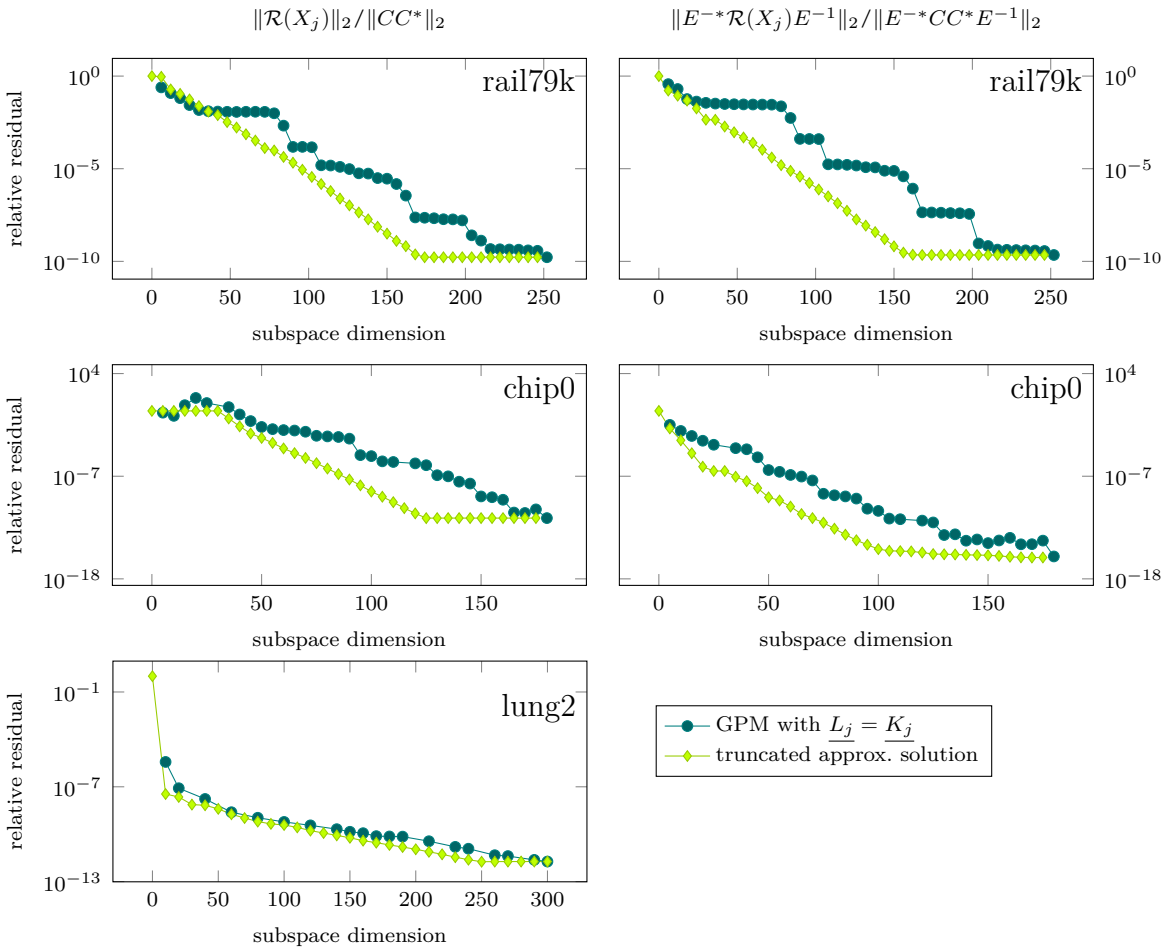

Figure 12.6.: Residual norms of orthogonal GPM iteration and of truncated approximate solutions.

and the residual norm of the final approximate solution, truncated with different values for r are displayed. In particular the rank r of the truncated approximate solution was increased by p from 0 to the maximum number of positive eigenvalues of $\underline{K_j} Y_j \underline{K_j}^*$. By construction $\tilde{Y}_j$ is a positive definite matrix and so we can calculate the Cholesky decomposition $\tilde{Y} = G^*G \in \mathbb{C}^{r\times r}$. Herewith the truncated approximate solution can be written as $\tilde{X}_j = (\tilde{Z}_j G^*)(G\tilde{Z}_j^*)$ with $\tilde{Z}_j = V_{j+1}\underline{\tilde{K}_j} \in \mathbb{C}^{n\times r}$. One sees that a significant rank reduction is possible in all examples with only little increase in the residual norm. The whole postprocessing procedure, including the eigenvalue decomposition and all residual norm calculations, is quite efficient. Again we have to distinguish the norm calculation of the residual for (10.1) and for (10.2). The former is more expensive due to

the multiplication with the system matrix E, see Section 11.4. For the rail79k example the process took 3.0 seconds (respectively 0.8 seconds), for the chip0 example we have 0.5 seconds (respectively 0.3 seconds) and for the lung2 example the procedure needed 0.9 seconds.

13. Conclusions

In this part a new ADI-type approach and a general projection framework for the approximation of complex large-scale algebraic Riccati equations were introduced. Rational Arnoldi decompositions were used to rewrite the Riccati residual. To obtain the Riccati ADI solution a rank condition was imposed on this formulation of the residual. A small-scale Lyapunov equation was obtained which characterizes the sought solution. It was shown that the Riccati ADI approximate solution exists and is unique under a simple condition for the shifts. Uniqueness of the solution implies the equivalence of our new approach and all previously known Riccati ADI methods.

We introduced two new iterative methods to calculate the Riccati ADI solution. Both make use of the fact that the solution of the small-scale Lyapunov equation can be updated efficiently when the rational Arnoldi decomposition is expanded. In our first iterative method, the Riccati RAD iteration, only system solves with matrices of the form $A^* - \mu I$ and the residual factor are necessary. The second method derived, the Lyapunov RADI iteration, contains the RADI iteration as a special case. Here, system solves with a matrix of the form $A^* - KB^* - \mu I$ and the residual factor are performed. In both algorithms the extension of the Krylov subspace was decoupled from the rest of the iteration, which made parallelization and, in case of real system matrices, realification possible easily.

We revealed that the Riccati ADI solution can be interpreted as an oblique projection onto a rational Krylov subspace if the kernel of the projection contains the Riccati residual factor. We derived an analytical expression for the residual factor, which is a rational function. Its poles and zeros were connected to the eigenvalues of projected system matrices.

A general projection framework was derived in which the Riccati residual is projected onto a (rational) Krylov subspace. The general projection method was formulated in the small-dimensional quantities given by the rational Arnoldi decomposition. It allows for the use of orthogonal and oblique projections which are handled in the same framework. No difference needs to be made for the use of polynomial, extended or rational Krylov subspaces. An efficient way to evaluate the residual norm was presented. It was shown that truncation of the obtained approximate solution can be interpreted as the solution of the Riccati residual projected to a subspace of the used Krylov subspace. Hence the residual norm evaluation for the truncated approximate solution can be handled with the efficient methods from the general projection method.

The numerical experiments show the competitiveness of our new ADI approach. Parallel system solves scale well with the number of parallel threads if they dominate the iteration, even though there seems to be a large overhead due to the use of MATLABs *parfor* command. The accuracy of the solution obtained with parallelization is remarkably good in all numerical examples, even for as many as 20 parallel solves. However, to make the parallel approach work in practice, a shift strategy has to be found which generates multiple shifts during the iteration. The general projection method with orthogonal projection showed the best convergence behavior among all other variants including the Riccati ADI iterates, but the projection methods clearly needed more time than the Riccati ADI iteration. Truncation of the GPM approximate solution proved to be an efficient way to reduce the rank of the approximate solution, i.e. the number of columns of the solution factor, while its residual norm is increased only slightly.

Part III.

An ODE framework for linear matrix equations

14. Introduction

A framework based on methods for numerical integration of ordinary differential equations (ODEs) for the solution of the Lyapunov equation

$$0 = \mathcal{L}(\mathcal{P}) \coloneqq A\mathcal{P} + \mathcal{P}A^{\mathsf{T}} + BB^{\mathsf{T}} \qquad \text{(1.2 revisited)}$$

with $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ is presented. In this part new insights into well-established methods are given. We assume that A is stable, so the unique symmetric positive definite solution of (1.2) is given by the controllability Gramian

$$\mathcal{P} = \int_0^\infty e^{At} BB^{\mathsf{T}} e^{A^{\mathsf{T}} t}\,\mathrm{d}t \in \mathbb{R}^{n\times n}, \qquad \text{(5.3 revisited)}$$

cf. Section 5. The integral representation of the solution makes an ODE based approach evident. The time dependent Gramian

$$P(t) = \int_0^t e^{A\tau} BB^{\mathsf{T}} e^{A^{\mathsf{T}}\tau} d\tau,$$

approximates the solution $\mathcal{P}$ of the Lyapunov equation for $t \to \infty$. We demonstrate that it is the solution of a system of ODEs which are then solved numerically with Runge-Kutta methods. We also consider the Sylvester equation

$$A\mathcal{Y} - \mathcal{Y}B - FG^{\mathsf{T}} = 0 \qquad \text{(2.2 revisited)}$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $F \in \mathbb{R}^{n\times r}$ and $G \in \mathbb{R}^{m\times r}$. We assume that the spectra of A and B are disjoint, so the solution $\mathcal{Y}$ is unique, cf. Section 2.5.

In particular, we will be concerned with (1.2) and (2.2) for large and sparse system matrices and a low rank initial residual, that is BB^T is of low rank for the Lyapunov equation and FG^T is of low rank for the Sylvester equation. The symmetric positive definite solution $\mathcal{P}$ of (1.2) is approximated by the low rank factorization $\mathcal{P} \approx ZZ^\mathsf{T}$ with the approximate Cholesky factors $Z \in \mathbb{C}^{n\times N}$ and $N \ll n$, [51, 6]. In a similar fashion the solution of the Sylvester equation is approximated by $\mathcal{Y} \approx \hat{Z}\Gamma\check{Z}^\mathsf{T}$ with rectangular matrices $\hat{Z}$, $\check{Z}$ consisting of N columns with $N \ll n, m$ and a diagonal matrix Γ, see, e.g. [14, 36].

In the following we give an overview of methods which are important or related to our later discussion. For a more exhaustive survey of methods for the solution of various linear matrix equations we refer to [63]. An important method is the ADI iteration for Lyapunov and Sylvester equations which was discussed in Section 4. Another type of methods for the solution of Lyapunov equations is making use of empirical Gramians [54]. The empirical Gramian essentially involves a sum approximation of the integral (5.3) $\mathcal{P} = \sum_j \delta_j g(t_j)$ for $g(t) = e^{At}BB^\mathsf{T}e^{A^\mathsf{T}t}$, arbitrary times t_j and appropriate quadrature weights δ_j. Usually, the identity $g(t) = h(t)h(t)^\mathsf{T}$ for $h(t) = e^{At}B$ is used to determine $g(t_j)$. In doing so, $h(t)$ is not computed directly, but as the solution of the ordinary differential equation (ODE)

$$\frac{\mathrm{d}}{\mathrm{d}t}h(t) = Ah(t) \tag{14.1}$$

with initial value $h(0) = B$. Any numerical integration scheme can be used to do so. Related quadrature approaches are discussed, e.g., in [61, 65]. Empirical Gramians are mainly used in model order reduction via balanced proper orthogonal decomposition (POD), see [60]. The ADI iteration and the quadrature-based methods have been connected to rational Krylov subspaces and moment matching, see [29, 69]. In [56] quadrature methods with complex time step sizes for the approximation of empirical Gramians were analyzed and connected to (rational) Krylov subspaces. Further the sta-

bility function of certain multi-stage implicit methods was connected to the (complex) interpolation points used in rational interpolation. In [72] shifted Legendre polynomials are used to obtain an approximate solution to the Lyapunov equation.

In this part the Gramian is approximated as the solution of a system of ODEs, where the ODE (14.1) for h and an ODE for $\mathcal{P}$ are solved numerically with Runge-Kutta methods. Using an ODE also for $\mathcal{P}$ gives us more flexibility than the sum approximation described above for the empirical Gramians. Our approach makes it possible to characterize the space spanned by the approximate solution and show equivalence of the ADI iteration and the use of particular Runge-Kutta methods.

The application of Runge-Kutta methods for the solution of a system of ODEs to approximate the Gramian is derived in Sections 15.1 and 15.2. We show in Section 15.3 that the approximate Cholesky factor Z of the resulting low rank approximation $\mathcal{P} \approx ZZ^*$ spans a rational Krylov subspace with poles depending on the eigenvalues of the used Butcher tableau and the time step sizes. An interpretation of balanced POD as a special case of our method is given in Section 15.4. In Section 15.5 we connect balancing related model order reduction using the ODE based Gramian approximations and rational interpolation. We explicitly state the interpolation points of the original and reduced LTI system.

The time dependent Gramian $P(t)$ does not satisfy the Lyapunov equation exactly. We observe that the residual possesses a low rank factorization $\mathcal{L}(P(t)) = h(t)h(t)^\mathsf{T}$ with $h(t) = e^{At}B$. In Section 16 we identify those Runge-Kutta methods which preserve this algebraic invariant, i.e. lead to iterates P_j that yield a low rank residual, too. For this purpose we use ideas from geometric numerical integration, where qualitative properties (e.g. algebraic invariants) of the solution are preserved instead of fulfilling quantitative properties (e.g. small errors), cf. [45, Ch. 5], [39]. In Section 16.1 we derive a residual based iteration which turns out to be equivalent to the ADI iteration and show how this iteration can be performed efficiently in Section 16.2. A realified variant of this iteration is derived in Section 16.3. The seemingly different approach to solve Lyapunov equations with shifted Legendre polynomials from [72] is connected to our geometric approach and

the ADI iteration in Section 16.4. Section 16.5 describes the necessary modifications for generalized Lyapunov equations. In Section 16.6 we demonstrate the effectiveness of our framework with numerical experiments. The invariant approach for Lyapunov equations is transferred to Sylvester equations in Section 17. We conclude in Section 18.

Most results of this part have already been published in [21] and [20]. The Gramian quadrature algorithm in Section 15 was first introduced in [21] and is stated here in the reworked variant of [20]. Section 15.2 extends the findings of [20, Sec. 2.2] with a realification approach. All results in the remaining subsections of Section 15 have been published in [20]. Section 16 is based on [21] but was adapted to the new formulation of the Gramian quadrature algorithm. In Section 16.1 the derivation of the multiplicative update formula was simplified. Moreover, the discussion is supplemented by a realification approach in Section 16.3. The findings of Section 17 have been published in [21, Sec. 4].

15. The Gramian quadrature algorithm

In this section we present a novel quadrature based algorithm to obtain approximate Cholesky factors of the Gramian. For reasons of simplicity we will not consider (1.2) in full generality, only matrices $B \in \mathbb{R}^{n\times 1}$ with one column will be considered. The case of a general matrix $B = [b_1, \dots, b_m] \in \mathbb{R}^{n\times m}$ can be reduced to

$$\begin{aligned}\mathcal{P} &= \int_0^\infty e^{At} B B^\mathsf{T} e^{A^\mathsf{T} t} \,\mathrm{d}t \\ &= \sum_{i=1}^m \int_0^\infty e^{At} b_i b_i^\mathsf{T} e^{A^\mathsf{T} t} \,\mathrm{d}t.\end{aligned}$$

Thus, our results extend to the case $m > 1$ easily.

Consider the system of ordinary differential equations

$$\frac{\mathrm{d}}{\mathrm{d}t} P(t) = h(t) h(t)^\mathsf{T}, \qquad P(0) = 0 \in \mathbb{R}^{n\times n}, \tag{15.1}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} h(t) = A h(t), \qquad h(0) = B \in \mathbb{R}^{n\times 1}. \tag{15.2}$$

Equation (15.2) is a linear, homogeneous differential equation which is solved by the function $h(t) = e^{At}B$. Due to the fundamental theorem of calculus equation (15.1) is solved by the time-dependent Gramian

$$P(t) = \int_0^t e^{A\tau} B B^{\mathsf{T}} e^{A^{\mathsf{T}}\tau} \,\mathrm{d}\tau = \int_0^t h(\tau) h(\tau)^{\mathsf{T}} \,\mathrm{d}\tau.$$

We intend to solve the above system of ODEs numerically to obtain an approximation to the Gramian $\mathcal{P} = \lim_{t\to\infty} P(t)$.

15.1. Approximating the Gramian via Runge-Kutta methods

The ODEs (15.1) and (15.2) are solved here with a partitioned Runge-Kutta method [39, Ch. II.2]: The ODE (15.1) is solved with a method based on a Butcher tableau with $\tilde{\Lambda} \in \mathbb{C}^{s\times s}$ and $\tilde{\beta} \in \mathbb{R}^s_{\geq 0}$. We only allow nonnegative real entries in $\tilde{\beta}$ to ensure that the approximation to the Gramian is positive semidefinite. The ODE (15.2) is solved using a Butcher tableau with $\Lambda \in \mathbb{C}^{s\times s}$ and $\beta \in \mathbb{C}^s$. Due to the complex valued matrices Λ, $\tilde{\Lambda}$ and β we make use of complex conjugated transposition $*$ instead of only transposition T in the following.

The Runge-Kutta iteration (6.2) is applied to the ODE (15.2), yielding the approximants $h_j \approx h(t_j)$. We obtain

$$h_j = h_{j-1} + \omega_j \sum_{i=1}^{s} \beta_i k_i^{(j)}, \qquad j = 1, \ldots, N, \tag{15.3}$$

with initial value $h_0 = B \in \mathbb{R}^{n\times 1}$. The slopes $k_i^{(j)}$ are given via (6.3) by

$$k_i^{(j)} = A\Big(h_{j-1} + \omega_j \sum_{\ell=1}^{s} \lambda_{i\ell} k_\ell^{(j)}\Big) \tag{15.4}$$

for $i = 1, \ldots, s$. Application of (6.2) to the ODE (15.1) yields the approximants $P_j \approx$

$P(t_j)$ via

$$P_j = P_{j-1} + \omega_j \sum_{i=1}^{s} \tilde{\beta}_i \tilde{k}_i^{(j)}, \qquad j = 1, \dots, N, \tag{15.5}$$

with initial value $P_0 = 0 \in \mathbb{R}^{n\times n}$. To obtain the slopes $\tilde{k}_i^{(j)}$ we apply (6.3) to (15.1). This yields $\tilde{k}_i^{(j)} = \mathfrak{h}_i^{(j)}(\mathfrak{h}_i^{(j)})^*$ with

$$\mathfrak{h}_i^{(j)} = h_{j-1} + \omega_j \sum_{\ell=1}^{s} \lambda_{i\ell} k_\ell^{(j)} \tag{15.6}$$

for $i = 1, \dots, s$ and with $k_\ell^{(j)}$ from (15.4) as the ODEs (15.1) and (15.2) are coupled (i.e. h from (15.2) appears in (15.1)).

We now aggregate the vectors $\mathfrak{h}_i^{(j)}$ and $k_i^{(j)}$ for $i = 1, \dots, s$ in matrices: Let $\mathcal{H}_j = [\mathfrak{h}_1^{(j)}, \dots, \mathfrak{h}_s^{(j)}] \in \mathbb{C}^{n\times s}$ and $K_j = [k_1^{(j)}, \dots, k_s^{(j)}] \in \mathbb{C}^{n\times s}$. Note that $k_i^{(j)} = A\mathfrak{h}_i^{(j)}$ holds. Thus

$$K_j = A\mathcal{H}_j. \tag{15.7}$$

To obtain $\mathcal{H}_j$ we rewrite (15.6) with the matrices K_j and $\mathcal{H}_j$

$$\begin{aligned} \mathcal{H}_j &= [h_{j-1}, \dots, h_{j-1}] + \omega_j K_j \Lambda^{\mathsf{T}} \\ &= \mathbb{1}_s^{\mathsf{T}} \otimes h_{j-1} + \omega_j A\mathcal{H}_j \Lambda^{\mathsf{T}}. \end{aligned} \tag{15.8}$$

In the latter equation $\mathcal{H}_j$ is the only unknown. In case $\mathcal{H}_j$ is computed from (15.8), we can determine K_j via (15.7).

Finally, using $\mathcal{H}_j$ and K_j, we express the sums in (15.3) and (15.5) through matrix multiplications. Herewith the iteration reads

$$\begin{aligned} P_j &= P_{j-1} + \mathcal{H}_j \operatorname{diag}(\omega_j \tilde{\beta}) \mathcal{H}_j^*, \\ h_j &= h_{j-1} + \omega_j K_j \beta = h_{j-1} + \omega_j A\mathcal{H}_j \beta. \end{aligned} \tag{15.9}$$

First $\mathcal{H}_j$ and K_j are determined using (15.8) and (15.7), then h_j and P_j are updated.

Remark 15.1. A different approach to obtain this iteration is via vectorization of the system of ODEs (15.1) and (15.2). This allows us to write the system in the form of (6.1)

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\mathrm{vec}(P(t))\\ h(t)\end{bmatrix} = \begin{bmatrix}\mathrm{vec}(h(t)h(t)^{\mathsf{T}})\\ Ah(t)\end{bmatrix} = f(t, y(t))$$

for the high-dimensional solution function $y\colon \mathbb{R} \to \mathbb{R}^{n^2+n}$, $y(t) = \begin{bmatrix}\mathrm{vec}(P(t))\\ h(t)\end{bmatrix}$, which needs to be determined. Clearly, the initial value reads $y(0) = \left[\begin{smallmatrix}0\\ B\end{smallmatrix}\right]$. This approach was used in [21], but can only be employed with one single Runge-Kutta method.

In order to see when $\mathcal{H}_j$ (and thus due to (15.7) also K_j) is uniquely determined, (15.8) is reformulated via vectorization as a linear system of equations with a system matrix of size $ns \times ns$

$$(I_{ns} - \omega_j(\Lambda \otimes A))\,\mathrm{vec}(\mathcal{H}_j) = h_{j-1} \otimes \mathbb{1}_s \in \mathbb{C}^{ns\times 1}. \tag{15.10}$$

Let $\mu_1, \ldots, \mu_s$ and $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of Λ and A respectively. Then the eigenvalues of $I_{ns} - \omega_j(\Lambda \otimes A)$ are given by $1 - \omega_j\mu_p\lambda_q$, $p = 1, \ldots, s$, $q = 1, \ldots, n$. Thus the solution of (15.10) is unique if and only if

$$\mu_p \neq \frac{1}{\omega_j\lambda_q} \tag{15.11}$$

for all $p = 1, \ldots, s$ and $q = 1, \ldots, n$.

The approximant P_j is by construction a positive semidefinite matrix and can be expressed as $P_j = Z_jZ_j^*$ for some complex valued matrix Z_j as $\tilde{\beta} \in \mathbb{R}^s_{\geq 0}$ holds. Thus we have

$$\begin{aligned} Z_jZ_j^* &= Z_{j-1}Z_{j-1}^* + \mathcal{H}_j\,\mathrm{diag}(\omega_j\tilde{\beta})\mathcal{H}_j^* \\ &= \left[Z_{j-1}, \mathcal{H}_j\,\mathrm{diag}(\omega_j\tilde{\beta})^{\frac{1}{2}}\right]\left[Z_{j-1}, \mathcal{H}_j\,\mathrm{diag}(\omega_j\tilde{\beta})^{\frac{1}{2}}\right]^*. \end{aligned} \tag{15.12}$$

Instead of iterating on P_j as in (15.9), the above observation allows us to iterate on the

Algorithm 15.1 Gramian quadrature algorithm

Input: $A \in \mathbb{R}^{n\times n}$ stable, $B \in \mathbb{R}^{n\times 1}$, positive time step sizes $\{\omega_1, \ldots, \omega_N\}$, Butcher tableau with $\tilde{\beta} \in \mathbb{R}^s_{\geq 0}$ and Butcher tableau with $\Lambda \in \mathbb{C}^{s\times s}$, $\beta \in \mathbb{C}^s$ which satisfies (15.11)

Output: $Z \in \mathbb{C}^{n\times sN}$ with $ZZ^* \approx \mathcal{P}$

1: initialize $h_0 = B$, $Z_0 = [\,]$
2: **for** $j = 1, \ldots, N$ **do**
3: solve $\mathcal{H}_j = [h_{j-1}, \ldots, h_{j-1}] + \omega_j A\mathcal{H}_j\Lambda^{\mathsf{T}}$ for $\mathcal{H}_j \in \mathbb{C}^{n\times s}$
4: update $Z_j = [Z_{j-1}, \mathcal{H}_j \operatorname{diag}(\omega_j\tilde{\beta})^{\frac{1}{2}}]$
5: $h_j = h_{j-1} + \omega_j A\mathcal{H}_j\beta$
6: **end for**
7: $Z = Z_N$

low rank factor

$$Z_j = [Z_{j-1}, \mathcal{H}_j \operatorname{diag}(\omega_j\tilde{\beta})^{\frac{1}{2}}] \in \mathbb{C}^{n\times js}$$

which gains s additional columns in every iteration step.

The procedure to obtain the Gramian approximation described in this section is summarized in Algorithm 15.1. We require that the eigenvalues of Λ satisfy (15.11) in order to ensure that all linear system solves have a unique solution and $\tilde{\beta} \in \mathbb{R}^s_{\geq 0}$ to ensure P_j is positive semidefinite.

15.2. Computation of $\mathcal{H}_j$

The most expensive part of Algorithm 15.1 is solving for $\mathcal{H}_j$ in line 3. Of course the linear system of equations (15.10) can be used to determine $\mathcal{H}_j$, but this approach is costly due to the dimension ns of the system matrix. We present a more efficient way to obtain $\mathcal{H}_j$ with the solution of s linear systems of dimension n. A related approach has been used in [26].

Let $(\Lambda')^{\mathsf{T}} = S\Lambda^{\mathsf{T}}S^{-1} \in \mathbb{C}^{s\times s}$ be a Schur decomposition of Λ^{T}, so the diagonal entries of the upper triangular matrix $(\Lambda')^{\mathsf{T}}$ are the eigenvalues $\mu_1, \ldots, \mu_s$ of Λ. Consider (15.8)

and define $\mathcal{H}'_j = [\mathfrak{h}'_1{}^{(j)}, \ldots, \mathfrak{h}'_s{}^{(j)}]$ via $\mathcal{H}_j = \mathcal{H}'_j S$. Then (15.8) can be reformulated as

$$\mathcal{H}'_j = (\mathbb{1}_s^{\mathsf{T}} \otimes h_{j-1})S^{-1} + \omega_j A \mathcal{H}'_j (\Lambda')^{\mathsf{T}}. \tag{15.13}$$

Let $[\alpha_1, \ldots, \alpha_s] = \mathbb{1}_s^{\mathsf{T}} S^{-1}$ be the row vector containing the column sums of S^{-1}. Then we can rewrite (15.13) as

$$\mathcal{H}'_j = [\alpha_1 h_{j-1}, \ldots, \alpha_s h_{j-1}] + \omega_j A \mathcal{H}'_j (\Lambda')^{\mathsf{T}}. \tag{15.14}$$

Looking only at the i-th column of (15.14) we obtain

$$\mathfrak{h}_i'^{(j)} = \alpha_i h_{j-1} + \omega_j \sum_{l=1}^{i-1} \lambda'_{il} A \mathfrak{h}_l'^{(j)} + \omega_j \mu_i A \mathfrak{h}_i'^{(j)},$$

so to calculate $\mathcal{H}'_j$ the following systems of linear equations have to be solved

$$(I_n - \omega_j \mu_i A)\mathfrak{h}_i'^{(j)} = \alpha_i h_{j-1} + \omega_j \sum_{l=1}^{i-1} \lambda'_{il} A \mathfrak{h}_l'^{(j)} \tag{15.15}$$

for $i = 1, \ldots, s$. Finally, $\mathcal{H}_j$ is assembled via $\mathcal{H}_j = \mathcal{H}'_j S$.

Assume that linear systems with a system matrix of dimension $\tau \times \tau$ are solved with a method needing $\mathcal{O}(\tau^3)$ flops. Then solving the ns-dimensional system (15.10) would need $\mathcal{O}(s^3 n^3)$ flops. In the procedure presented here a Schur decomposition of the $s \times s$ matrix Λ is necessary to obtain (15.15), at the costs of $\mathcal{O}(s^3)$ flops. To solve the s systems of dimension $n \times n$ in (15.15) further $\mathcal{O}(sn^3)$ flops are necessary. All in all the costs are reduced from $\mathcal{O}(s^3 n^3)$ flops to only $\mathcal{O}(sn^3) + \mathcal{O}(s^3)$ flops.

Finally let us consider the case in which all quantities are real valued. Even then the diagonal entries of $(\Lambda')^{\mathsf{T}}$ might be complex and thus introduce complex variables into the iteration. To avoid this we use the real Schur decomposition for $(\Lambda')^{\mathsf{T}}$, in which the

diagonal consists of real 1×1 or 2×2 blocks. Assume that

$$(\Lambda'_{i,i+1})^{\mathsf{T}} = \begin{bmatrix} \lambda'_{i,i} & \lambda'_{i+1,i} \\ \lambda'_{i,i+1} & \lambda'_{i+1,i+1} \end{bmatrix}$$

is such a 2×2 diagonal block corresponding to the eigenvalues μ, $\overline{\mu}$ with $\mathrm{Im}(\mu) \neq 0$. Looking at column i and $i+1$ of (15.14) we then find

$$\left[\mathfrak{h}_i'^{(j)}, \mathfrak{h}_{i+1}'^{(j)}\right] = [z_1, z_2] + \omega_j A \left[\mathfrak{h}_i'^{(j)}, \mathfrak{h}_{i+1}'^{(j)}\right] (\Lambda'_{i,i+1})^{\mathsf{T}} \tag{15.16}$$

with

$$\begin{aligned} z_1 &= \alpha_i h_{j-1} + \omega_j \sum_{l=1}^{i-1} \lambda'_{i,l} A \mathfrak{h}_l'^{(j)}, \\ z_2 &= \alpha_{i+1} h_{j-1} + \omega_j \sum_{l=1}^{i-1} \lambda'_{i+1,l} A \mathfrak{h}_l'^{(j)}. \end{aligned} \tag{15.17}$$

Vectorization of (15.16) yields the real $2n$-dimensional linear system

$$\begin{bmatrix} I - \omega_j \lambda'_{i,i} A & -\omega_j \lambda'_{i,i+1} A \\ -\omega_j \lambda'_{i+1,i} A & I - \omega_j \lambda'_{i+1,i+1} A \end{bmatrix} \begin{bmatrix} \mathfrak{h}_i'^{(j)} \\ \mathfrak{h}_{i+1}'^{(j)} \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}. \tag{15.18}$$

We further modify this procedure to obtain $\mathfrak{h}_i'^{(j)}$ and $\mathfrak{h}_{i+1}'^{(j)}$ such that only the solution of one n-dimensional complex system is necessary. Then linear combinations of the real and imaginary parts of the solution are used, similar as in Section 9.3. For this we utilize yet another similarity transformation $T \in \mathbb{R}^{2\times 2}$ which fulfills

$$T(\Lambda'_{i,i+1})^{\mathsf{T}} T^{-1} = \begin{bmatrix} \mathrm{Re}(\mu) & \mathrm{Im}(\mu) \\ -\mathrm{Im}(\mu) & \mathrm{Re}(\mu) \end{bmatrix}.$$

Such a transformation matrix T can be obtained as a solution of the homogeneous

Sylvester equation

$$\begin{bmatrix} \operatorname{Re}(\mu) & \operatorname{Im}(\mu) \\ -\operatorname{Im}(\mu) & \operatorname{Re}(\mu) \end{bmatrix} T - T(\Lambda'_{i,i+1})^{\mathsf{T}} = 0, \tag{15.19}$$

e.g. via vectorization. We find that (15.16) is equivalent to

$$[v_1,\, v_2] = [y_1,\, y_2] + \omega_j A[v_1,\, v_2] \begin{bmatrix} \operatorname{Re}(\mu) & \operatorname{Im}(\mu) \\ -\operatorname{Im}(\mu) & \operatorname{Re}(\mu) \end{bmatrix}$$

with $[v_1,\, v_2] = [\mathfrak{h}_i^{\prime(j)},\, \mathfrak{h}_{i+1}^{\prime(j)}]T^{-1}$ and $[y_1,\, y_2] = [z_1,\, z_2]T^{-1}$. Again via vectorization we obtain the $2n$-dimensional system

$$\begin{bmatrix} I - \omega_j \operatorname{Re}(\mu) A & \omega_j \operatorname{Im}(\mu) A \\ -\omega_j \operatorname{Im}(\mu) A & I - \omega_j \operatorname{Re}(\mu) A \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}.$$

This linear system is in turn equivalent to the n-dimensional complex system

$$(I_n - \omega_j \mu A)(v_1 + \imath v_2) = y_1 + \imath y_2, \tag{15.20}$$

which can be easily verified by looking at the real and imaginary part of (15.20) separately. All in all, to calculate $[\mathfrak{h}_i^{\prime(j)},\, \mathfrak{h}_{i+1}^{\prime(j)}]$, one has to obtain T from (15.19), solve the n-dimensional complex system (15.20) and then set $[\mathfrak{h}_i^{\prime(j)},\, \mathfrak{h}_{i+1}^{\prime(j)}] = [v_1,\, v_2]T$. This procedure turned out to be faster than solving the $2n$-dimensional real system (15.18) in numerical experiments. Thus, instead of solving two complex $n \times n$ systems in Algorithm 16.1 for the steps with $\Lambda^{(j)} = \mu$ and $\Lambda^{(j+1)} = \overline{\mu}$ it is possible to solve one real $2n \times 2n$ system (15.18). As the system matrix as well as the right-hand side is real, this gives $\mathcal{H}_j \in \mathbb{R}^{n \times 2}$.

The realification approaches described above are comparable to realification methods for the ADI iteration presented in [48], namely the methods $\mathsf{M2}^*, \mathsf{M4}^*$ (solution of a real $2n$-dimensional system) and $\mathsf{M4}$ (solution of a complex n-dimensional system, summarized in Section 4). There, too, it was found that the approach $\mathsf{M4}$ with the complex

n-dimensional system is the most efficient one, see [48, Ch. 4.1.5].

15.3. The space spanned by the approximate Cholesky factor

In this section the space spanned by the approximate solution generated by the Gramian quadrature algorithm is characterized. The main result is that the columns of the approximate Cholesky factor $Z = Z_N$ obtained from Algorithm 15.1 span a (rational) Krylov subspace which is essentially determined by the eigenvalues of $\omega_i \Lambda$. To show this we first demonstrate how the iterate Z can be obtained in only one step of Algorithm 15.1 with a Butcher tableau assembled from Λ, β, $\tilde{\beta}$ and the time step sizes ω_j.

The approximate Cholesky factor Z is recursively defined via line 4 of Algorithm 15.1. Expanding the for loop we find that after N steps

$$Z = [\mathcal{H}_1, \ldots, \mathcal{H}_N] \begin{bmatrix} \operatorname{diag}(\omega_1 \tilde{\beta})^{\frac{1}{2}} & & \\ & \ddots & \\ & & \operatorname{diag}(\omega_N \tilde{\beta})^{\frac{1}{2}} \end{bmatrix} \tag{15.21}$$

is obtained. To characterize $[\mathcal{H}_1, \ldots, \mathcal{H}_N]$ we expand line 3 and line 5 from Algorithm 15.1. For $\mathcal{H}_1$ we find

$$\begin{aligned} \mathcal{H}_1 &= \mathbb{1}_s^{\mathsf{T}} \otimes h_0 + \omega_1 A \mathcal{H}_1 \Lambda^{\mathsf{T}} \\ &= \mathbb{1}_s^{\mathsf{T}} \otimes h_0 + A \mathcal{H}_1 (\omega_1 \Lambda^{\mathsf{T}}). \end{aligned} \tag{15.22}$$

For $\mathcal{H}_2$ we obtain

$$\begin{aligned}\mathcal{H}_2 &= \mathbb{1}_s^\mathsf{T} \otimes h_1 + \omega_2 A \mathcal{H}_2 \Lambda^\mathsf{T} \\ &= \mathbb{1}_s^\mathsf{T} \otimes (h_0 + \omega_1 A \mathcal{H}_1 \beta) + \omega_2 A \mathcal{H}_2 \Lambda^\mathsf{T} \\ &= \mathbb{1}_s^\mathsf{T} \otimes h_0 + A\mathcal{H}_1(\omega_1[\beta, \ldots, \beta]) + A\mathcal{H}_2(\omega_2 \Lambda^\mathsf{T}) \\ &= \mathbb{1}_s^\mathsf{T} \otimes h_0 + A[\mathcal{H}_1, \mathcal{H}_2] \begin{bmatrix} \omega_1[\beta, \ldots, \beta] \\ \omega_2 \Lambda^\mathsf{T} \end{bmatrix}.\end{aligned} \tag{15.23}$$

Putting $\mathcal{H}_1$ from (15.22) and $\mathcal{H}_2$ from (15.23) together, one yields

$$[\mathcal{H}_1, \mathcal{H}_2] = \mathbb{1}_{2s}^\mathsf{T} \otimes h_0 + A[\mathcal{H}_1, \mathcal{H}_2] \begin{bmatrix} \omega_1 \Lambda^\mathsf{T} & \omega_1[\beta, \ldots, \beta] \\ 0 & \omega_2 \Lambda^\mathsf{T} \end{bmatrix}.$$

Proceeding in this way up to iteration step N and setting $\hat{\mathcal{H}} = [\mathcal{H}_1, \ldots, \mathcal{H}_N]$ this leads to the equation

$$\hat{\mathcal{H}} = \mathbb{1}_{Ns}^\mathsf{T} \otimes h_0 + A \hat{\mathcal{H}} \hat{\Lambda}^\mathsf{T} \tag{15.24}$$

with the assembled matrix

$$\hat{\Lambda}^\mathsf{T} := \begin{bmatrix} \omega_1 \Lambda^\mathsf{T} & \omega_1[\beta, \ldots, \beta] & \cdots & \omega_1[\beta, \ldots, \beta] \\ 0 & \omega_2 \Lambda^\mathsf{T} & \omega_2[\beta, \ldots, \beta] & \omega_2[\beta, \ldots, \beta] \\ \vdots & 0 & \ddots & \vdots \\ 0 & \cdots & 0 & \omega_N \Lambda^\mathsf{T} \end{bmatrix} \in \mathbb{C}^{Ns \times Ns}. \tag{15.25}$$

The above derivations show that Z from (15.21) can also be interpreted as one step of Algorithm 15.1 with time step size $\boldsymbol{\omega}_1 = 1$, $\boldsymbol{\beta} = [\omega_1 \beta^\mathsf{T}, \ldots, \omega_N \beta^\mathsf{T}]^\mathsf{T}$, $\tilde{\boldsymbol{\beta}} = [\omega_1 \tilde{\beta}^\mathsf{T}, \ldots, \omega_N \tilde{\beta}^\mathsf{T}]^\mathsf{T}$ and $\boldsymbol{\Lambda} = \hat{\Lambda}$ from (15.25). It is therefore sufficient to characterize the space spanned by Z after one step of Algorithm 15.1 as the situation with more steps is contained as a special case.

Next (15.24) is transformed so that the space spanned by Z can be identified. Let all

entries of $\tilde{\beta}$ be positive, i.e. $\tilde{\beta} \in \mathbb{R}^s_+$, then the diagonal matrix in (15.21) is regular and so the space spanned by the columns of Z equals the one spanned by the columns of $\hat{\mathcal{H}}$. We proceed with similarity transformations of $\hat{\Lambda}^\mathsf{T}$ as in Section 15.2 to uncouple the columns of $\hat{\mathcal{H}}$. Define $\hat{\mathcal{H}} = \hat{\mathcal{H}}' S$ with a similarity transformation $S \in \mathbb{C}^{Ns \times Ns}$ which transforms $\hat{\Lambda}^\mathsf{T}$ to its Jordan canonical form

$$(\hat{\Lambda}')^\mathsf{T} = S\hat{\Lambda}^\mathsf{T}S^{-1} = \begin{bmatrix} J_1 & & \\ & \ddots & \\ & & J_q \end{bmatrix} \tag{15.26}$$

with q Jordan blocks $J_l \in \mathbb{C}^{s_l \times s_l}$ of dimension s_l for $l = 1, \ldots, q$. We further partition $\hat{\mathcal{H}}' = [\hat{\mathcal{H}}'_1, \ldots, \hat{\mathcal{H}}'_q]$ and

$$\mathbb{1}_{Ns}^\mathsf{T} S^{-1} = [\alpha^{(1)}, \ldots, \alpha^{(q)}] \tag{15.27}$$

according to the sizes of the Jordan blocks, i.e. $\hat{\mathcal{H}}'_l \in \mathbb{C}^{n \times s_l}$ and $(\alpha^{(l)})^\mathsf{T} \in \mathbb{C}^{s_l}$. Multiplication of (15.24) with S^{-1} from the right yields

$$[\hat{\mathcal{H}}'_1, \ldots, \hat{\mathcal{H}}'_q] = [\alpha^{(1)}, \ldots, \alpha^{(q)}] \otimes h_0 + A[\hat{\mathcal{H}}'_1, \ldots, \hat{\mathcal{H}}'_q] \begin{bmatrix} J_1 & & \\ & \ddots & \\ & & J_q \end{bmatrix}.$$

Due to the partitioning this equation is equivalent to

$$\hat{\mathcal{H}}'_l = \alpha^{(l)} \otimes h_0 + A\hat{\mathcal{H}}'_l J_l \quad \text{for } l = 1, \ldots, q.$$

The matrices $\hat{\mathcal{H}}'_l = [\hat{\mathfrak{h}}_1^{\prime(l)}, \ldots, \hat{\mathfrak{h}}_{s_l}^{\prime(l)}]$ are determined by

$$\begin{aligned} (I_n - \hat{\mu}_l A)\hat{\mathfrak{h}}_1^{\prime(l)} &= \alpha_1^{(l)} h_0, \\ (I_n - \hat{\mu}_l A)\hat{\mathfrak{h}}_i^{\prime(l)} &= \alpha_i^{(l)} h_0 + A\hat{\mathfrak{h}}_{i-1}^{\prime(l)} \quad \text{for } i = 2, \ldots, s_l \end{aligned} \tag{15.28}$$

with the eigenvalue $\hat{\mu}_l$ of $\hat{\Lambda}^\mathsf{T}$ as the diagonal element of the Jordan block J_l.

We next state a technical lemma which is needed in the proof of the main result of this section.

Lemma 15.2. *Let* $(\mathbb{1}_{Ns}^{\mathsf{T}}, \hat{\Lambda}^{\mathsf{T}})$ *be observable. Then there exists a transformation matrix* S *to Jordan canonical form in* (15.26) *such that* $\alpha^{(l)} = [1, 0, \dots, 0]$ *holds for* $l = 1, \dots, q$ *in* (15.27).

Proof. For $l = 1, \dots, q$ define $e_l = [1, 0 \dots, 0]^{\mathsf{T}} \in \mathbb{R}^{s_l \times 1}$. Assume that there exist polynomials $\mathfrak{p}_l$ with

$$\alpha^{(l)} = e_l^{\mathsf{T}} \mathfrak{p}_l(J_l). \tag{15.29}$$

Now replace the matrix S in (15.26) and (15.27) with $\tilde{S} = \operatorname{diag}(\mathfrak{p}_1(J_1), \dots, \mathfrak{p}_q(J_q))S$. As J_l commutes with rational functions in J_l the matrix $\tilde{S}$ is a similarity transformation to Jordan canonical form, too, and it holds

$$\begin{aligned}
\mathbb{1}_{Ns}^{\mathsf{T}} \tilde{S}^{-1} &= \mathbb{1}_{Ns}^{\mathsf{T}} S^{-1} \operatorname{diag}(\mathfrak{p}_1(J_1), \dots, \mathfrak{p}_q(J_q))^{-1} \\
&= [\alpha^{(1)}, \dots, \alpha^{(q)}] \operatorname{diag}(\mathfrak{p}_1(J_1)^{-1}, \dots, \mathfrak{p}_q(J_q)^{-1}) \\
&= [e_1^{\mathsf{T}}, \dots, e_q^{\mathsf{T}}].
\end{aligned}$$

It remains to show that a polynomial $\mathfrak{p}_l$ fulfilling (15.29) exists and $\mathfrak{p}_l(J_l)$ is invertible for $l = 1, \dots, q$. Define the upper shift matrix $\mathfrak{r}_l(J_l) = -\hat{\mu}_l I + J_l$ with ones above the diagonal and zeros everywhere else. It holds $e_l^{\mathsf{T}} \mathfrak{r}_l(J_l)^{i-1} = [0, \dots, 0, 1, 0, \dots, 0]$, a vector with a one at position i for $i = 1, \dots, s_l$. For the ith row of $\mathfrak{p}_l(J_l)$ we find with (15.29)

$$\begin{aligned}
[0, \dots, 0, 1, 0, \dots 0] \mathfrak{p}_l(J_l) &= e_l^{\mathsf{T}} \mathfrak{r}_l(J_l)^{i-1} \mathfrak{p}_l(J_l) \\
&= e_l^{\mathsf{T}} \mathfrak{p}_l(J_l) \mathfrak{r}_l(J_l)^{i-1} \\
&= \alpha^{(l)} \mathfrak{r}_l(J_l)^{i-1} \\
&= [0, \dots, 0, \alpha_1^{(l)}, \dots, \alpha_{s_l-(i-1)}^{(l)}].
\end{aligned}$$

This implies that $\mathfrak{p}_l(J_l)$ is an upper triangular matrix with entries $\alpha_1^{(l)}$ on the diagonal.

As $(\mathbb{1}_{Ns}^{\mathsf{T}}, \hat{\Lambda}^{\mathsf{T}})$ is observable, so is $(\alpha^{(l)}, J_l)$ and thus $\alpha_1^{(l)} \neq 0$. So $\mathfrak{p}_l(J_l)$ is invertible, which concludes the proof. □

We note that the observability condition for $(\mathbb{1}_{Ns}^{\mathsf{T}}, \hat{\Lambda}^{\mathsf{T}})$ (and so for $(\alpha^{(l)}, J_l)$) implies that there exists only one Jordan block per eigenvalue of $\hat{\Lambda}^{\mathsf{T}}$, i.e. $\hat{\Lambda}^{\mathsf{T}}$ is a non-derogatory matrix. These preparations allow us to state the following theorem, which characterizes the Krylov subspace spanned by the approximate solution generated by the Gramian quadrature algorithm.

Theorem 15.3. *Let $Ns < n$ and assume that $(\mathbb{1}_{Ns}^{\mathsf{T}}, \hat{\Lambda}^{\mathsf{T}})$ is observable. Let $\hat{\Lambda}^{\mathsf{T}}$ have eigenvalues $\hat{\mu}_1, \dots, \hat{\mu}_q$. If $\hat{\mu}_l \neq 0$ then*

$$\begin{aligned}\operatorname{span}(\hat{\mathcal{H}}_l') &= \operatorname{span}\Big\{(I_n - \hat{\mu}_l A)^{-i} h_0 \mid i = 1, \dots, s_l\Big\} \\ &= \operatorname{span}\Big\{(A - \hat{\mu}_l^{-1} I_n)^{-i} h_0 \mid i = 1, \dots, s_l\Big\}.\end{aligned}$$

If $\hat{\mu}_l = 0$ then

$$\operatorname{span}(\hat{\mathcal{H}}_l') = \operatorname{span}\Big\{A^i h_0 \mid i = 0, \dots, s_l - 1\Big\}.$$

Proof. Set $\hat{\mathfrak{h}}_i' := \hat{\mathfrak{h}}_i'^{(l)}$ for better readability. Due to the observability of $(\mathbb{1}_{Ns}^{\mathsf{T}}, \hat{\Lambda}^{\mathsf{T}})$ we find from (15.26) and (15.27) that $(\alpha^{(l)}, J_l)$ is observable. Due to Lemma 15.2 we can assume $\alpha^{(l)} = [1, 0, \dots, 0]$.

Let $\hat{\mu}_l \neq 0$. Because of (15.28)

$$\operatorname{span}(\hat{\mathfrak{h}}_1') = \operatorname{span}((I_n - \hat{\mu}_l A)^{-1} h_0)$$

holds. From (15.28) we find for $1 < i \leq s_l$ as $\alpha_i^{(l)} = 0$

$$\begin{aligned}\hat{\mathfrak{H}}'_i &= (I_n - \hat{\mu}_l A)^{-1} A \hat{\mathfrak{H}}'_{i-1} \\ &= (I_n - \hat{\mu}_l A)^{-1}(-\hat{\mu}_l^{-1}(I_n - \hat{\mu}_l A) + \hat{\mu}_l^{-1} I_n)\hat{\mathfrak{H}}'_{i-1} \\ &= -\hat{\mu}_l^{-1}\hat{\mathfrak{H}}'_{i-1} + \hat{\mu}_l^{-1}(I_n - \hat{\mu}_l A)^{-1}\hat{\mathfrak{H}}'_{i-1} \\ &= -\hat{\mu}_l^{-1}\hat{\mathfrak{H}}'_{i-1} - (A - \hat{\mu}_l^{-1} I_n)^{-1}\hat{\mathfrak{H}}'_{i-1}.\end{aligned}$$

Via induction this concludes the first part of the proof.

Now let $\hat{\mu}_l = 0$. From (15.28)

$$\operatorname{span}(\hat{\mathfrak{H}}'_1) = \operatorname{span} h_0$$

is immediate. For $1 < i \leq s_l$ we have

$$\hat{\mathfrak{H}}'_i = A\hat{\mathfrak{H}}'_{i-1},$$

and the claim again results from induction. □

The above lemma show that the space spanned by $\hat{\mathcal{H}} = [\hat{\mathcal{H}}'_1, \ldots, \hat{\mathcal{H}}'_q]S$ is a Krylov subspaces. From Algorithm 15.1 we have $h_0 = B$ and so it holds $\operatorname{span}(\hat{\mathcal{H}}) = \mathcal{K}(A, B, s)$, where s is the set of general eigenvalues of the matrix pencil $(I_{Ns}, \hat{\Lambda}^\mathsf{T})$. Even when in Theorem 15.3 the observability of $(\mathbb{1}_{Ns}^\mathsf{T}, \hat{\Lambda}^\mathsf{T})$ is omitted, then still Krylov subspaces with the same poles are spanned by $\hat{\mathcal{H}}$, but with lower multiplicity.

Remark 15.4. We can reformulate (15.24) to obtain a RKD-like decomposition for $\hat{\mathcal{H}}$

$$A\begin{bmatrix} B & \hat{\mathcal{H}} \end{bmatrix}\begin{bmatrix} 0 \\ \hat{\Lambda}^\mathsf{T} \end{bmatrix} = \begin{bmatrix} B & \hat{\mathcal{H}} \end{bmatrix}\begin{bmatrix} -\mathbb{1}_{Ns}^\mathsf{T} \\ I_{Ns} \end{bmatrix}.$$

The above decomposition needs not be a rational Krylov decomposition as it may not meet the requirements of Definition 3.3. Consider e.g. the case where $B \in \operatorname{span}(\hat{\mathcal{H}})$, then $[B, \hat{\mathcal{H}}]$ is not of full rank. If this decomposition is a RKD we the poles of the Krylov

subspace spanned by $\hat{\mathcal{H}}$ are the eigenvalues of $(I_{Ns}, \hat{\Lambda}^{\mathsf{T}})$.

15.4. Connection to balanced POD

We now show the connection of the Gramian quadrature algorithm presented in Section 15.1 to balanced POD, another method involving Gramian approximations with low rank Cholesky factors. We only consider the controllability Gramian $\mathcal{P}$ here as the approximation of the observability Gramian $\mathcal{Q}$ is done analogously.

In the method balanced proper orthogonal decomposition of snapshots (BPOD) as discussed in [60], the Gramians are approximated with finite sums (see [67] for a related approach). In particular the controllability Gramian is approximated via

$$\begin{aligned}\mathcal{P} = \int_0^\infty h(t)h(t)^{\mathsf{T}}\,\mathrm{d}t &\approx \int_0^T h(t)h(t)^{\mathsf{T}}\,\mathrm{d}t \\ &\approx \sum_{i=1}^{N} \delta_i h_i h_i^*, \end{aligned} \tag{15.30}$$

with $h_i \approx h(t_i)$, $h(t) = e^{At}B$, an end time $T \in \mathbb{R}_+$, times $t_1 < \cdots < t_N \in [0, T]$ and quadrature weights δ_i. The approximation of $h(t_i)$ is done by solving the ODE

$$\frac{\mathrm{d}}{\mathrm{d}t}h(t) = Ah(t), \qquad h(0) = B. \tag{15.31}$$

From (15.30) we find that the approximate Cholesky factor is given by

$$Z_{\mathrm{c}} = [h_1, \ldots, h_N]\,\mathrm{diag}\left(\sqrt{\delta_1}, \ldots, \sqrt{\delta_N}\right).$$

A central task in BPOD is the numerical solution of the ODE (15.31). Unfortunately it is not stated which numerical method should be used for solving the ODE. In the following we assume a Runge-Kutta method with Λ_h and β_h is used to solve the ODE in the same way as (15.2) was solved in Section 15.1. In particular, for $h_0 = B$ and time

step sizes $\omega_j = t_j - t_{j-1}$ this means

$$\begin{aligned} \mathcal{H}_j &= [h_{j-1}, \dots, h_{j-1}] + \omega_j A \mathcal{H}_j \Lambda_h^{\mathsf{T}} \\ h_j &= h_{j-1} + \omega_j A \mathcal{H}_j \beta_h^{\mathsf{T}} \end{aligned} \tag{15.32}$$

just as in Algorithm 15.1, but in the BPOD method the approximate Cholesky factor is updated via

$$Z_j = [Z_{j-1}, h_j \delta_j^{\frac{1}{2}}]$$

instead of $Z_j = [Z_{j-1}, \mathcal{H}_j \operatorname{diag}(\omega_j \tilde{\beta})^{\frac{1}{2}}]$ as in Algorithm 15.1. We illustrate how the balanced POD iterates can be obtained by Algorithm 15.1 in the case where $h_j \delta_j h_j^*$ and $\mathcal{H}_j \operatorname{diag}(\omega_j \tilde{\beta}) \mathcal{H}_j^*$ coincide. Due to the dimension of h_j and $\mathcal{H}_j$ this is only possible for Butcher tableaus of size $s = 1$ or for $\tilde{\beta}$ having only one nonzero entry.

We first consider the case $s = 1$ and thus have $\mathcal{H}_j \in \mathbb{C}^{n \times 1}$. So (15.32) becomes

$$\begin{aligned} \mathcal{H}_j &= h_{j-1} + \omega_j A \mathcal{H}_j \Lambda_h^{\mathsf{T}} \\ h_j &= h_{j-1} + \omega_j A \mathcal{H}_j \beta_h^{\mathsf{T}}, \end{aligned}$$

i.e. $\mathcal{H}_j = h_j$ if $\Lambda_h = \beta_h$. This is e.g. fulfilled in the backward Euler method with $\Lambda_h = \beta_h = 1$. If additionally $\tilde{\beta} = \delta_j / \omega_j$, balanced POD and Algorithm 15.1 produce the same iterates.

In case of arbitrary Butcher tableaus with s-dimensional Λ_h and β_h the way BPOD fits into the framework presented here is rather crude. Consider a Butcher tableau with the $(s+1)$-dimensional matrices

$$\Lambda = \begin{bmatrix} \Lambda_h & 0 \\ \beta_h^{\mathsf{T}} & 0 \end{bmatrix}, \ \beta = \begin{bmatrix} \beta_h \\ 0 \end{bmatrix}, \ \tilde{\beta} = \begin{bmatrix} 0 \\ \delta_j / \omega_j \end{bmatrix}.$$

Algorithm 15.1 generates the iterate

$$\underbrace{[\mathfrak{h}_1^{(j)}, \ldots, \mathfrak{h}_s^{(j)}, \mathfrak{h}_{s+1}^{(j)}]}_{=\mathcal{H}_j} = [h_{j-1}, \ldots, h_{j-1}] + \omega_j A[\mathfrak{h}_1^{(j)}, \ldots, \mathfrak{h}_s^{(j)}, \mathfrak{h}_{s+1}^{(j)}] \begin{bmatrix} \Lambda_h^\mathsf{T} & \beta_h \\ 0 & 0 \end{bmatrix}.$$

Separating the first s columns from the last one yields

$$\begin{aligned} [\mathfrak{h}_1^{(j)}, \ldots, \mathfrak{h}_s^{(j)}] &= [h_{j-1}, \ldots, h_{j-1}] + \omega_j A[\mathfrak{h}_1^{(j)}, \ldots, \mathfrak{h}_s^{(j)}]\Lambda_h^\mathsf{T} \\ \mathfrak{h}_{s+1}^{(j)} &= h_{j-1} + \omega_j A[\mathfrak{h}_1^{(j)}, \ldots, \mathfrak{h}_s^{(j)}]\beta_h \end{aligned}$$

and so $h_j = \mathfrak{h}_{s+1}$. Due to the zero entries in $\tilde{\beta}$ we further find

$$\begin{aligned} \mathcal{H}_j \operatorname{diag}(\omega_j \tilde{\beta}) \mathcal{H}_j^* &= \mathfrak{h}_j \omega_j \frac{\delta_j}{\omega_j} \mathfrak{h}_j^* \\ &= h_j \delta_j h_j^* \end{aligned}$$

i.e. Algorithm 15.1 and BPOD produce the same iterates for this special choice of tableaus.

15.5. Application to model order reduction

We now use the Gramian approximations of the previous sections to generate an approximate balancing transformation for model order reduction as described in Section 5.1. It was shown in [60, Prop. 2] that if approximate Cholesky factors with $\operatorname{rank}(Z_\mathrm{o}^\mathsf{T} Z_\mathrm{c}) = r$ are used in balanced truncation, then the projection matrix V from (5.4) contains the first columns of an approximate balancing transformation. In [56] it was found that for certain quadrature methods for solving (14.1) the reduced system obtained by balanced POD matches some moments. We proceed as in balanced POD to obtain an approximate balancing transformation. For the calculation of the approximate Cholesky factors Algorithm 15.1 is used. The procedure is summarized in Algorithm 15.2. It allows us to identify a connection between the Butcher tableau which characterizes the Runge-Kutta method and the expansion points at which moments are

Algorithm 15.2 Approximate balancing transformation

Input: system matrices $A \in \mathbb{R}^{n\times n}$ stable, $B \in \mathbb{R}^{n\times 1}$, $C \in \mathbb{R}^{1\times n}$, positive time step sizes $\{\omega_1, \ldots, \omega_N\}$ and $\{\tau_1, \ldots, \tau_N\}$, Butcher tableaus with $\tilde{\beta}_\mathrm{c}, \tilde{\beta}_\mathrm{o} \in \mathbb{R}^s_{\geq 0}$ and Butcher tableaus with $\Lambda_\mathrm{c}, \Lambda_\mathrm{o} \in \mathbb{C}^{s\times s}$, $\beta_\mathrm{c}, \beta_\mathrm{o} \in \mathbb{C}^s$ which satisfy (15.11)

Output: reduced system matrices $\hat{A} \in \mathbb{C}^{r\times r}$, $\hat{B} \in \mathbb{C}^{r\times 1}$, $\hat{C} \in \mathbb{C}^{1\times r}$ with $r = \operatorname{rank}(Z_\mathrm{o}^* Z_\mathrm{c})$

1: obtain Z_c with $Z_\mathrm{c} Z_\mathrm{c}^* \approx \mathcal{P}$ from Algorithm 15.1 with A, B, Λ_c, β_c, $\tilde{\beta}_\mathrm{c}$ and $\{\omega_1, \ldots, \omega_N\}$

2: obtain Z_o with $Z_\mathrm{o} Z_\mathrm{o}^* \approx \mathcal{Q}$ from Algorithm 15.1 with A^T, C^T, Λ_o, β_o, $\tilde{\beta}_\mathrm{o}$ and $\{\tau_1, \ldots, \tau_N\}$

3: calculate compact SVD $Z_\mathrm{o}^* Z_\mathrm{c} = U \Sigma T^*$

4: assemble projection matrices $V = Z_\mathrm{c} T \Sigma^{-\frac{1}{2}}$, $W = Z_\mathrm{o} U \Sigma^{-\frac{1}{2}}$

5: return $\hat{A} = W^* A V$, $\hat{B} = W^* B$, $\hat{C} = C V$

matched. Note that due to the use of Butcher tableaus with complex entries in general complex reduced system matrices are obtained. This is the reason for using conjugate transposition $*$ instead of transposition T.

As will be shown next, the transfer function of the reduced system generated by Algorithm 15.2 interpolates the transfer function of the original system at expansion points which depend on the eigenvalues of the Butcher tableaus and the time step sizes. In particular the expansion points are the inverse eigenvalues of $\omega_i \Lambda_\mathrm{c}$ for $i = 1, \ldots, N_\mathrm{c}$ and the conjugated inverse eigenvalues of $\tau_i \Lambda_\mathrm{o}$ for $i = 1, \ldots, N_\mathrm{o}$.

Theorem 15.5. *Let the inputs of Algorithm 15.2 with $\tilde{\beta}_c, \tilde{\beta}_o \in \mathbb{R}^s_+$ be given. Define $\hat{\Lambda}_c^\mathsf{T}$ as in* (15.25) *with Λ_c, β_c and $\{\omega_1, \ldots, \omega_N\}$. Define $\hat{\Lambda}_o^\mathsf{T}$ as in* (15.25) *with Λ_o, β_o and $\{\tau_1, \ldots, \tau_N\}$. Let $\{\hat{\mu}_1, \ldots, \hat{\mu}_{q_c}\} = \cup_{i=1}^N \lambda(\omega_i \Lambda_c)$ and $\{\hat{\nu}_1, \ldots, \hat{\nu}_{q_o}\} = \cup_{i=1}^N \lambda(\tau_i \Lambda_o)$ be the eigenvalues of $\hat{\Lambda}_c$ and $\hat{\Lambda}_o$ with multiplicities $s_1, \ldots, s_{q_c}$ and $t_1, \ldots, t_{q_o}$.*

If $(\mathbb{1}_{Ns}^\mathsf{T}, \hat{\Lambda}_c^\mathsf{T})$ and $(\mathbb{1}_{Ns}^\mathsf{T}, \hat{\Lambda}_o^\mathsf{T})$ are observable and $\operatorname{rank} Z_o^* Z_c = Ns$ holds, then the transfer function of the reduced system with system matrices $\hat{A}, \hat{B}, \hat{C}$ produced by Algorithm 15.2 satisfies

$$\begin{aligned} \hat{G}^{(i)}(\hat{\mu}_{l_c}^{-1}) &= G^{(i)}(\hat{\mu}_{l_c}^{-1}) \quad \text{for } i = 0, \ldots, s_{l_c} - 1, \\ \hat{G}^{(i)}(\overline{\hat{\nu}}_{l_o}^{-1}) &= G^{(i)}(\overline{\hat{\nu}}_{l_o}^{-1}) \quad \text{for } i = 0, \ldots, t_{l_o} - 1 \end{aligned} \tag{15.33}$$

for $l_c = 1, \ldots, q_c$ and $l_o = 1, \ldots, q_o$. For any zero eigenvalues the corresponding inter-

polation in (15.33) *has to be read as interpolation at* ∞*. If some of the values* $\hat{\mu}_i$ *and* $\overline{\hat{\nu}_j}$ *coincide, even higher derivatives of the transfer function are interpolated.*

Proof. The reduced system is generated via projection with the matrices V and W. Due to line 3 and line 4 of Algorithm 15.2 and as $Z_{\mathrm{o}}^* Z_{\mathrm{c}}$ is regular $\operatorname{span}(V) = \operatorname{span}(Z_{\mathrm{c}})$ and $\operatorname{span}(W) = \operatorname{span}(Z_{\mathrm{o}})$ hold. With Theorem 15.3 we find for $\hat{\mu}_{l_{\mathrm{c}}}, \hat{\nu}_{l_{\mathrm{o}}} \neq 0$

$$\operatorname{span}\Big\{(A - \hat{\mu}_{l_{\mathrm{c}}}^{-1} I_n)^{-i} B \mid i = 1, \ldots, s_{l_{\mathrm{c}}}\Big\} \subseteq \operatorname{span}(V),$$
$$\operatorname{span}\Big\{(A^{\mathsf{T}} - \hat{\nu}_{l_{\mathrm{o}}}^{-1} I_n)^{-i} C^{\mathsf{T}} \mid i = 1, \ldots, t_{l_{\mathrm{o}}}\Big\} \subseteq \operatorname{span}(W).$$

Further, if $\hat{\mu}_{l_{\mathrm{c}}}, \hat{\nu}_{l_{\mathrm{o}}} = 0$, then

$$\operatorname{span}\Big\{A^i B \mid i = 0, \ldots, s_{l_{\mathrm{c}}} - 1\Big\} \subseteq \operatorname{span}(V),$$
$$\operatorname{span}\Big\{(A^{\mathsf{T}})^i C^{\mathsf{T}} \mid i = 0, \ldots, t_{l_{\mathrm{o}}} - 1\Big\} \subseteq \operatorname{span}(W).$$

Due to Section 5.2 this concludes the proof. □

It is interesting to see that using a Runge-Kutta method it is not possible to match moments around the expansion point zero, as this would require an infinite eigenvalue of Λ from the Butcher tableau or an infinite time step size, which is impossible.

We note that in [56] complex time step sizes ω_j (τ_j respectively) are used in Runge-Kutta methods to achieve moment matching around complex expansion points. This is unfeasible in the method presented here as then the iterates P_j are in general not positive semidefinite and the approximate Cholesky factors Z_j would not exist. Instead, in the framework presented here, complex tableaus may be used.

Let us illustrate the moment matching property from Theorem 15.5 in some examples. We state the expansion points at which moments are matched for certain Runge-Kutta methods and visualize them in the complex plane.

Explicit Runge-Kutta methods are parameterized by Butcher tableaus with strictly lower triangular Λ. As such matrices have just zero eigenvalues only moments around ∞ are matched for explicit methods. An example is Euler's method given by the Butcher tableau with $\Lambda = 0$, $\beta = 1$.

For the backward Euler method we have $\Lambda = 1$, $\beta = 1$, so the moments are matched around the inverse time step sizes ω_j^{-1} and τ_j^{-1}.

Consider the Butcher tableaus from the Gauß-Legendre and Radau IA method of size $s = 2$. The Gauß-Legendre method [45, Ch. 3.4] is given by

$$\Lambda_{\mathrm{GL}} = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} - \frac{1}{6}\sqrt{3} \\ \frac{1}{4} + \frac{1}{6}\sqrt{3} & \frac{1}{4} \end{bmatrix}, \ \beta_{\mathrm{GL}} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \end{bmatrix}.$$

This method is equivalent to the Hammer-Hollingsworth method [24] which was used in [56]. The matrix Λ_{GL} has eigenvalues $\mu_{1/2} = \frac{1}{4} \pm \frac{\sqrt{3}}{12} i$. The Radau IA method [25, Sec. 344] is given by

$$\Lambda_{\mathrm{R}} = \begin{bmatrix} \frac{1}{4} & -\frac{1}{4} \\ \frac{1}{4} & \frac{5}{12} \end{bmatrix}, \ \beta_{\mathrm{R}} = \begin{bmatrix} \frac{1}{4} \\ \frac{3}{4} \end{bmatrix}.$$

It has eigenvalues $\lambda_{1/2} = \frac{1}{3} \pm \frac{\sqrt{2}}{6} i$.

When Algorithm 15.2 is executed with the Gauß-Legendre method for Z_{c} and the Radau IA method for Z_{o}, then the moments are matched around the expansion points

$$(\omega_j \mu_{1/2})^{-1} = \omega_j^{-1}(3 \mp \sqrt{3} i) \ \text{ and } \ (\tau_j \lambda_{1/2})^{-1} = \tau_j^{-1}(2 \mp \sqrt{2} i)$$

for $j = 1, \ldots, N$. These expansion points are visualized in the complex plane in Figure 15.1 for $\omega_j = \tau_j = 0.3, 0.4, \ldots, 1$.

16. Preservation of an algebraic invariant

In the previous section the Gramian quadrature algorithm was presented. We now identify particular Runge-Kutta methods which preserve an algebraic invariant, namely the rank of the initial residual. Therefore conditions for the corresponding Butcher tableaus are derived. It is shown that the algorithm for these special Runge-Kutta methods is equivalent to the ADI iteration. Throughout this section we assume that both

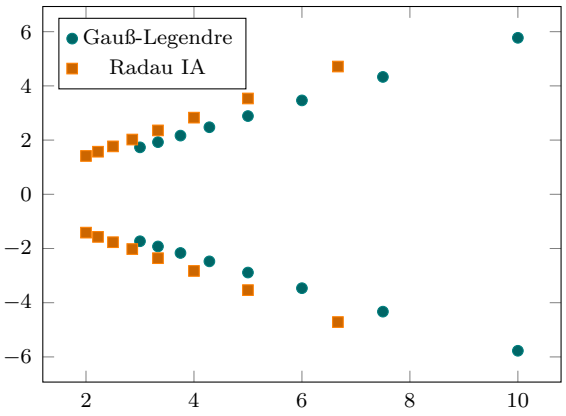

Figure 15.1.: Expansion points in the complex plane for the Gauß-Legendre and Radau IA methods.

ODEs (15.1) and (15.2) are solved numerically with the same Runge-Kutta method, i.e. that $\tilde{\beta} = \beta$ holds in Algorithm 15.1.

First, observe that the time dependent Gramian $P(t)$ does not satisfy the Lyapunov equation (1.2) exactly for finite times $t \in \mathbb{R}$. The Lyapunov residual

$$\mathcal{L}(P(t)) = AP(t) + P(t)A^\mathsf{T} + BB^\mathsf{T}$$

remains. Next, reconsider the derivative of $P(t)$ in (15.1)

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}P(t) &= h(t)h(t)^\mathsf{T} \\
&= h(0)h(0)^\mathsf{T} + \int_0^t \frac{\mathrm{d}}{\mathrm{d}\tau}h(\tau)h(\tau)^\mathsf{T}\,\mathrm{d}\tau \\
&= BB^\mathsf{T} + \int_0^t \left(Ah(\tau)h(\tau)^\mathsf{T} + h(\tau)h(\tau)^\mathsf{T}A^\mathsf{T}\right)\mathrm{d}\tau \\
&= BB^\mathsf{T} + A\int_0^t h(\tau)h(\tau)^\mathsf{T}\,\mathrm{d}\tau + \int_0^t h(\tau)h(\tau)^\mathsf{T}\,\mathrm{d}\tau A^\mathsf{T} \\
&= BB^\mathsf{T} + AP(t) + P(t)A^\mathsf{T}.
\end{aligned}$$

Thus we have

$$\mathcal{L}(P(t)) = AP(t) + P(t)A^\mathsf{T} + BB^\mathsf{T} = h(t)h(t)^\mathsf{T} \tag{16.1}$$

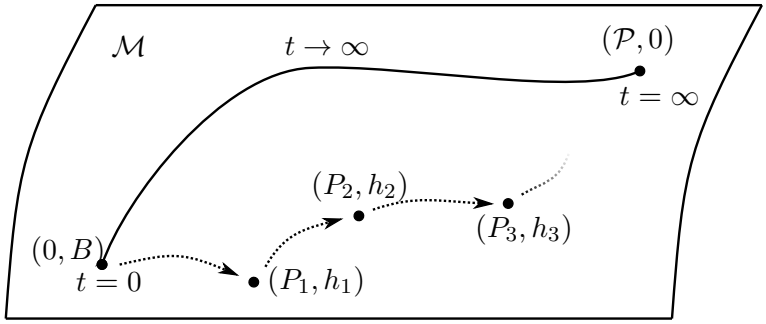

Figure 16.1.: Solution of $\dot{h} = Ah$, $\dot{P} = hh^{\mathsf{T}}$ and iterates from Algorithm 15.1 with tableaus satisfying Theorem 16.1 evolving on the rank-one residual manifold $\mathcal{M}$.

for all t. Obviously, due to the right-hand side, the Lyapunov residual is of rank one.

We intend to use only those Butcher tableaus which guarantee that the iterates $P_j = Z_j Z_j^*$ and h_j satisfy the algebraic invariant (16.1) in the sense

$$AP_j + P_j A^{\mathsf{T}} + BB^{\mathsf{T}} = h_j h_j^*. \tag{16.2}$$

Please note that as the tableaus might be complex valued also the iterates P_j and h_j may be complex valued and thus we need to modify T in (16.1) to $*$ in (16.2). Note also that due to Theorem 15.3 and with $\beta \in \mathbb{R}_+$ the space spanned by Z_j is a Krylov subspace, so the approximate solution P_j is an ADI approximate solution when (16.1) is satisfied, as then the rank of the Lyapunov residual is one.

Before we discuss which Butcher tableaus allow for (16.2) let us give an interpretation of (16.1) and (16.2). Consider all tuples (P, h), for which the invariant (16.2) is satisfied. They are located on the manifold

$$\mathcal{M} := \left\{ (P, h) \text{ with } P \in \mathbb{C}^{n \times n},\ h \in \mathbb{C}^{n \times 1} \mid AP + PA^{\mathsf{T}} + BB^{\mathsf{T}} - hh^* = 0 \right\}. \tag{16.3}$$

The solution $(P(t), h(t))$ of (15.1) and (15.2) lies on $\mathcal{M}$ for all times $t \in \mathbb{R}$ because of (16.1). As the Lyapunov residual (16.1) is of rank one we will call $\mathcal{M}$ the rank-one residual manifold. The time dependent Gramian evolves on the rank-one residual manifold with $h(t) \to 0$ for $t \to \infty$; see Fig. 16.1. Enforcing (16.2) for the iterates (P_j, h_j), in general we obtain P_j which do not approximate the trajectory of the time

dependent Gramian $P(t)$ but which are located on $\mathcal{M}$. Therefore the approximation $P_N \approx \mathcal{P}$ is good when the iterate h_N is small, because then the tuple $(P_N, h_N) \in \mathcal{M}$ is located close to $(\mathcal{P}, 0) \in \mathcal{M}$.

We now show for which Runge-Kutta methods the iterates in Algorithm 15.1 lie on the rank-one residual manifold.

Theorem 16.1. *Let $A \in \mathbb{R}^{n\times n}$ be stable, $B \in \mathbb{R}^n$ and $\omega_i > 0$, $i = 1, \dots, j$. Consider a Butcher tableau* (6.4) *with $\Lambda \in \mathbb{C}^{s\times s}$ and $\tilde{\beta} = \beta \in \mathbb{R}^s_+$ satisfying* (15.11)*. After j steps of Algorithm 15.1 we find*

$$\begin{aligned} &AP_j + P_jA^\mathsf{T} + BB^\mathsf{T} \\ &\qquad = h_jh_j^* + \sum_{i=1}^{j} \omega_i^2 K_i \left(\operatorname{diag}(\beta)\overline{\Lambda} + \Lambda^\mathsf{T}\operatorname{diag}(\beta) - \beta\beta^\mathsf{T}\right) K_i^* \end{aligned} \tag{16.4}$$

for $P_j = Z_jZ_j^*$. Thus, if

$$\operatorname{diag}(\beta)\overline{\Lambda} + \Lambda^\mathsf{T}\operatorname{diag}(\beta) - \beta\beta^\mathsf{T} = 0 \tag{16.5}$$

holds, the iterates P_j and h_j satisfy the invariance equation (16.2)*.*

Proof. For $j = 0$ the statement is obviously true as $P_0 = Z_0Z_0^\mathsf{T} = 0$ and $h_0 = B$.

For $j \in \mathbb{N}$ we first use (15.12) and then inductively (16.4) for $j-1$

$$\begin{aligned} &AP_j + P_jA^\mathsf{T} + BB^\mathsf{T} \\ &= AP_{j-1} + P_{j-1}A^\mathsf{T} + BB^\mathsf{T} + \omega_jA\mathcal{H}_j\operatorname{diag}(\beta)\mathcal{H}_j^* + \omega_j\mathcal{H}_j\operatorname{diag}(\beta)\mathcal{H}_j^*A^\mathsf{T} \\ &= h_{j-1}h_{j-1}^* + \sum_{i=1}^{j-1} \omega_i^2 K_i \left(\operatorname{diag}(\beta)\overline{\Lambda} + (\operatorname{diag}(\beta)\Lambda)^\mathsf{T} - \beta\beta^\mathsf{T}\right) K_i^* \\ &\qquad + \omega_jA\mathcal{H}_j\operatorname{diag}(\beta)\mathcal{H}_j^* + \omega_j\mathcal{H}_j\operatorname{diag}(\beta)\mathcal{H}_j^*A^\mathsf{T}. \end{aligned} \tag{16.6}$$

Using $A\mathcal{H}_j = K_j$ and expanding $\mathcal{H}_j$ as in (15.8) we obtain

$$\begin{aligned} A\mathcal{H}_j\operatorname{diag}(\beta)\mathcal{H}_j^* &= K_j\operatorname{diag}(\beta)\left([h_{j-1}, \dots, h_{j-1}] + \omega_jK_j\Lambda^\mathsf{T}\right)^* \\ &= K_j\beta h_{j-1}^* + \omega_jK_j\operatorname{diag}(\beta)\overline{\Lambda}K_j^*. \end{aligned}$$

Inserting this in (16.6) and adding a zero we find

$$\begin{aligned}
&AP_j + P_jA^\mathsf{T} + BB^\mathsf{T} \\
&= h_{j-1}h_{j-1}^* + \sum_{i=1}^{j-1} \omega_i^2 K_i \left(\operatorname{diag}(\beta)\overline{\Lambda} + (\operatorname{diag}(\beta)\Lambda)^\mathsf{T} - \beta\beta^\mathsf{T}\right) K_i^* \\
&\quad + \omega_j K_j \beta h_{j-1}^* + \omega_j^2 K_j \operatorname{diag}(\beta)\overline{\Lambda} K_j^* + \omega_j h_{j-1}\beta^\mathsf{T} K_j^* + \omega_j^2 K_j \Lambda^\mathsf{T} \operatorname{diag}(\beta) K_j^* \\
&\quad + \omega_j^2 K_j \beta\beta^\mathsf{T} K_j^* - \omega_j^2 K_j \beta\beta^\mathsf{T} K_j^* \\
&= h_{j-1}h_{j-1}^* + \omega_j K_j \beta h_{j-1}^* + \omega_j h_{j-1}\beta^\mathsf{T} K_j^* + \omega_j^2 K_j \beta\beta^\mathsf{T} K_j^* \\
&\quad + \omega_j^2 K_j \left(\operatorname{diag}(\beta)\overline{\Lambda} + \Lambda^\mathsf{T} \operatorname{diag}(\beta) - \beta\beta^\mathsf{T}\right) K_j^* \\
&\quad + \sum_{i=1}^{j-1} \omega_i^2 K_i \left(\operatorname{diag}(\beta)\overline{\Lambda} + (\operatorname{diag}(\beta)\Lambda)^\mathsf{T} - \beta\beta^\mathsf{T}\right) K_i^* \\
&= h_j h_j^* + \sum_{i=1}^{j} \omega_i^2 K_i \left(\operatorname{diag}(\beta)\overline{\Lambda} + (\operatorname{diag}(\beta)\Lambda)^\mathsf{T} - \beta\beta^\mathsf{T}\right) K_i^*,
\end{aligned}$$

as $h_j = h_{j-1} + \omega_j K_j \beta$. This proves (16.4) and concludes the proof. □

The next question is if Runge-Kutta methods exist which fulfill (16.5). Indeed there are numerous such Butcher tableaus and we present a systematic way to construct suitable tableaus. First, observe that the equation $\operatorname{diag}(\beta)\overline{\Lambda} + (\operatorname{diag}(\beta)\Lambda)^\mathsf{T} - \beta\beta^\mathsf{T} = 0$ implies that the diagonal entries satisfy

$$\beta_j\overline{\lambda_{jj}} + \beta_j\lambda_{jj} - \beta_j^2 = \beta_j\left(2\operatorname{Re}(\lambda_{jj}) - \beta_j\right) = 0$$

for $j = 1, \ldots, s$. Thus either $\beta_j = 0$, or

$$\beta_j = 2\operatorname{Re}(\lambda_{jj}), \qquad j = 1, \ldots, s.$$

As $\beta_j \in \mathbb{R}_+$ is required, this implies that the diagonal elements of Λ have to be in $\mathbb{C}_+$ whenever (16.5) is required.

The simplest 1-stage Butcher tableaus satisfying (16.5) are given for an arbitrary

(complex) number $\mu \in \mathbb{C}_+$ by

$$\Lambda = \mu,\ \beta = 2\,\mathrm{Re}(\mu).$$

The implicit midpoint rule

$$\Lambda = \frac{1}{2},\ \beta = 1$$

is a prominent example of such a 1-stage tableau. A simple 2-stage tableau which satisfies (16.5) is the 2-stage implicit Runge-Kutta method

$$\Lambda = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} - \frac{1}{6}\sqrt{3} \\ \frac{1}{4} + \frac{1}{6}\sqrt{3} & \frac{1}{4} \end{bmatrix},\ \beta = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \end{bmatrix}.$$

The latter two methods belong to the family of Gauß-Legendre methods which are special s-stage implicit Runge-Kutta methods based on Gauß-Legendre quadrature [45, Ch. 3.4]. For $s \in \mathbb{N}$, the respective method is unique and satisfies (16.5), see [45, Lemma 5.3] and the subsequent corollary, where the matrix M in [45] corresponds to the left-hand side of (16.5).

Another family of methods for which (16.5) holds is given by DIRK methods of the form

$$\Lambda = \begin{bmatrix} \mu_1 & 0 & 0 & \cdots & \cdots & 0 \\ 2\,\mathrm{Re}(\mu_1) & \mu_2 & 0 & \cdots & \cdots & 0 \\ 2\,\mathrm{Re}(\mu_1) & 2\,\mathrm{Re}(\mu_2) & \mu_3 & \ddots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & \mu_{s-1} & 0 \\ 2\,\mathrm{Re}(\mu_1) & 2\,\mathrm{Re}(\mu_2) & \cdots & \cdots & 2\,\mathrm{Re}(\mu_{s-1}) & \mu_s \end{bmatrix},\ \beta = \begin{bmatrix} 2\,\mathrm{Re}(\mu_1) \\ 2\,\mathrm{Re}(\mu_2) \\ \vdots \\ 2\,\mathrm{Re}(\mu_s) \end{bmatrix} \tag{16.7}$$

with $\mu_1, \ldots, \mu_s \in \mathbb{C}_+$. In this case, it is easy to verify that the necessary and sufficient condition (15.11)

$$\mu_p \neq \frac{1}{\omega_j \lambda_q} = \frac{1}{\omega_j |\lambda_q|^2} \overline{\lambda_q},$$

is satisfied for all $p = 1, \dots, s$ and $q = 1, \dots, n$: As λ_q is an eigenvalue of the stable matrix A, we have $\mathrm{Re}(\overline{\lambda_q}) < 0$. Thus, for a stable A any DIRK method with a tableau (16.7) satisfies (15.11).

16.1. A multiplicative update formula for the residual factor h_j

The vector h_j is the residual factor for methods which satisfy the invariance equation (16.5). We derive a multiplicative update formula for these iterates which is connected to the stability function of the corresponding Runge-Kutta method. To do so we take a new perspective on the Runge-Kutta methods. So far we have used the same tableau with Λ, β in every iteration step. The time step sizes ω_j for $j = 1, \dots, N$ may vary. In the following we allow for varying tableaus with $\Lambda^{(j)} \in \mathbb{C}^{s\times s}$, $\beta^{(j)} \in \mathbb{C}^s$ during the iteration, in particular the matrices $\Lambda^{(j)}$ do not need to have the same eigenvalues. This implies the iteration

$$\begin{aligned} y_j &= y_{j-1} + \omega_j \sum_{i=1}^{s} \beta_i^{(j)} k_i^{(j)}, && j = 1, \dots, N, \\ k_i^{(j)} &= f\Big(t_{j-1} + \gamma_i^{(j)},\, y_{j-1} + \omega_j \sum_{\ell=1}^{s} \lambda_{i\ell}^{(j)} k_\ell^{(j)}\Big), && i = 1, \dots, s, \end{aligned}$$

instead of (6.2) and (6.3). Further we observe that every $\Lambda^{(j)}$ and $\beta^{(j)}$ is multiplied with the time step size ω_j. This allows us to fix the time step sizes at $\omega_j = 1$ by replacing $\Lambda^{(j)}$ and $\beta^{(j)}$ by $\omega_j\Lambda^{(j)}$ and $\omega_j\beta^{(j)}$. Algorithm 15.1 has to be modified accordingly. In line 3 the term $\omega_j\Lambda$ has to be replaced by $\Lambda^{(j)}$, while in line 4 and 5 the terms $\omega_j\tilde{\beta}$ and $\omega_j\beta$ have to be replaced by $\tilde{\beta}^{(j)}$ and $\beta^{(j)}$. Apparently Theorem 16.1 remains true even when different tableaus are used in every step as long as (16.5) holds for $\Lambda^{(i)}$ and $\beta^{(i)}$, $i = 1, \dots, j$.

We now focus on the iterates $h_j = h_{j-1} + A\mathcal{H}_j\beta^{(j)}$ as computed in line 5 of Algorithm 15.1 (modified as described above) from Butcher tableaus with $\Lambda^{(j)} \in \mathbb{C}^{s\times s}$ and $\beta^{(j)} \in \mathbb{C}^s$, where $\mathcal{H}_j$ is as in (15.8). Our goal is to rewrite the update rule as a multiplicative one, i.e. we want to find a matrix M_j with $h_j = M_j h_{j-1}$. We use the transpose of (15.8)

(modified as described above)

$$\begin{aligned} \mathcal{H}_j^\mathsf{T} &= h_{j-1}^\mathsf{T} \otimes \mathbb{1}_s + \Lambda^{(j)} \mathcal{H}_j^\mathsf{T} A^\mathsf{T} \\ \Leftrightarrow \mathcal{H}_j^\mathsf{T} - \Lambda^{(j)} \mathcal{H}_j^\mathsf{T} A^\mathsf{T} &= h_{j-1}^\mathsf{T} \otimes \mathbb{1}_s \end{aligned}$$

and vectorize to find

$$\operatorname{vec}(\mathcal{H}_j^\mathsf{T}) = (I_{ns} - A \otimes (\Lambda^{(j)}))^{-1} (I_n \otimes \mathbb{1}_s) h_{j-1}$$

with $\operatorname{vec}(h_{j-1}^\mathsf{T} \otimes \mathbb{1}_s) = \operatorname{vec}(\mathbb{1}_s h_{j-1}^\mathsf{T} I_n)$. Again via vectorization we obtain

$$\begin{aligned} A\mathcal{H}_j\beta^{(j)} &= \operatorname{vec}((\beta^{(j)})^\mathsf{T} \mathcal{H}_j^\mathsf{T} A^\mathsf{T}) \\ &= (A \otimes (\beta^{(j)})^\mathsf{T}) \operatorname{vec}(\mathcal{H}_j^\mathsf{T}) \\ &= (A \otimes (\beta^{(j)})^\mathsf{T}) \left(I_{ns} - A \otimes \Lambda^{(j)}\right)^{-1} (I_n \otimes \mathbb{1}_s) h_{j-1}. \end{aligned}$$

Thus the iterate h_j is obtained from h_{j-1} via

$$h_j = h_{j-1} + A\mathcal{H}_j\beta^{(j)} = M_j h_{j-1} \tag{16.8}$$

with the iteration matrix

$$M_j \coloneqq M_j(A) = I_n + (A \otimes (\beta^{(j)})^\mathsf{T}) \left(I_{ns} - A \otimes \Lambda^{(j)}\right)^{-1} (I_n \otimes \mathbb{1}_s). \tag{16.9}$$

The matrix-valued function $M_j(z)$ is a generalization of the stability function $R_{(j)}(z) = 1 + z(\beta^{(j)})^\mathsf{T}(I_s - z\Lambda^{(j)})^{-1}\mathbb{1}_s$ of the corresponding Runge-Kutta method.

Next the iteration matrix (16.9) is transformed to see that it is determined by the stability function of the used Runge-Kutta method. Let the system matrix $A = VDV^{-1}$ be diagonalizable with the matrix of right eigenvectors V and the diagonal matrix $D = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ containing the eigenvalues of A on the diagonal. Then the iteration

matrix M_j from (16.9) simplifies considerably and reads

$$\begin{aligned}
M_j &= I_n + (A \otimes (\beta^{(j)})^{\mathsf{T}}) \left(I_{ns} - (A \otimes \Lambda^{(j)})\right)^{-1} (I_n \otimes \mathbb{1}_s) \\
&= I_n + V(D \otimes (\beta^{(j)})^{\mathsf{T}})(V^{-1} \otimes I_s) \\
&\qquad \cdot \left\{I_n \otimes I_s - (V \otimes I_s)(D \otimes \Lambda^{(j)})(V^{-1} \otimes I_s)\right\}^{-1} (I_n \otimes \mathbb{1}_s) \\
&= I_n + V(D \otimes (\beta^{(j)})^{\mathsf{T}}) \left(I_n \otimes I_s - (D \otimes \Lambda^{(j)})\right)^{-1} (I_n \otimes \mathbb{1}_s) V^{-1} \\
&= V \left(I_n + \begin{bmatrix} \lambda_1(\beta^{(j)})^{\mathsf{T}} & & \\ & \ddots & \\ & & \lambda_n(\beta^{(j)})^{\mathsf{T}} \end{bmatrix} \begin{bmatrix} I_s - \lambda_1 \Lambda^{(j)} & & \\ & \ddots & \\ & & I_s - \lambda_n \Lambda^{(j)} \end{bmatrix}^{-1} \begin{bmatrix} \mathbb{1}_s & & \\ & \ddots & \\ & & \mathbb{1}_s \end{bmatrix} \right) V^{-1} \\
&= V \begin{bmatrix} 1 + \lambda_1(\beta^{(j)})^{\mathsf{T}}(I_s - \lambda_1 \Lambda^{(j)})^{-1} \mathbb{1}_s & & \\ & \ddots & \\ & & 1 + \lambda_n(\beta^{(j)})^{\mathsf{T}}(I_s - \lambda_n \Lambda^{(j)})^{-1} \mathbb{1}_s \end{bmatrix} V^{-1} \\
&= V \begin{bmatrix} R_{(j)}(\lambda_1) & & \\ & \ddots & \\ & & R_{(j)}(\lambda_n) \end{bmatrix} V^{-1}, \qquad (16.10)
\end{aligned}$$

i.e. the iteration matrix is determined by the stability function $R_{(j)}(z)$ corresponding to the Butcher tableau with $\Lambda^{(j)}$, $\beta^{(j)}$ and the eigenvalues and eigenvectors of the system matrix A. For non-diagonalizable system matrices we have no explicit formula in terms of the stability function. However, M_j can be interpreted as a composition of continuous functions. As such it depends continuously an A. The diagonalizable matrices are dense in the set of all matrices. Thus M_j is (implicitly) determined by the stability function of the utilized Runge-Kutta method for non-diagonalizable matrices A, too. We conclude that the iterate h_j is the same for various tableaus, as long as the stability functions coincide.

16.2. Efficient iteration on the rank-one residual manifold

Assume that (16.5) is satisfied by the tableaus, i.e. the iterates lie on the rank-1 residual manifold. Then not only h_j but also the iterate $P_j = Z_j Z_j^*$ (but not Z_j) is determined by h_j via (16.2). Thus, tableaus with the same stability function yield the same approximation $P_j = Z_j Z_j^*$, as long as the tableaus satisfy (15.11), (16.5) and

$\beta \in \mathbb{R}_+^s$, which we assume in the rest of this section. Next we investigate when two such s-stage Runge-Kutta methods have the same stability function. Assume that Λ and β are as in Theorem 16.1 and satisfy (16.5), $0 = \operatorname{diag}(\beta)\overline{\Lambda} + \Lambda^{\mathsf{T}} \operatorname{diag}(\beta) - \beta\beta^{\mathsf{T}}$. Then

$$-\overline{\Lambda} = \operatorname{diag}(\beta)^{-1}(\Lambda^{\mathsf{T}} - \beta \mathbb{1}_s^{\mathsf{T}}) \operatorname{diag}(\beta)$$

as $\beta_j > 0$, $j = 1, \ldots, s$, proving that the matrices $-\overline{\Lambda}$ and $\Lambda - \mathbb{1}_s\beta^{\mathsf{T}}$ are similar. Let Λ have eigenvalues $\mu_1, \ldots, \mu_s$. Then with the determinant based characterization of the stability function (6.5) we find that

$$\begin{aligned} R(z) &= \frac{\det(I + z(\mathbb{1}_s\beta^{\mathsf{T}} - \Lambda))}{\det(I - z\Lambda)} = \frac{\det(I + z\overline{\Lambda})}{\det(I - z\Lambda)} \\ &= \frac{(1 + \overline{\mu_1}z) \cdots (1 + \overline{\mu_s}z)}{(1 - \mu_1 z) \cdots (1 - \mu_s z)} \end{aligned} \tag{16.11}$$

holds. This is just the stability function of a DIRK method as given in (16.7). Thus any method based on a Butcher tableau with Λ, β such that Λ has the eigenvalues $\mu_1, \ldots, \mu_s$ and $\beta \in \mathbb{R}_+^s$ satisfies (16.5) is equivalent to a DIRK method (16.7). Note that obviously the order of the parameters μ_i is irrelevant.

Due to (16.8) we see that $h_N = \left(\prod_{j=1}^N M_j\right) h_0$ holds. From (16.10) we find that hence the final approximation $P = Z_N Z_N^*$ is determined by the product of the stability functions of the Runge-Kutta methods used during the iteration. Let $\Lambda^{(j)}$ and $\beta^{(j)} \in \mathbb{R}_+^{s_j}$ satisfy (16.5) for $j = 1, \ldots, N$ with the eigenvalues $\mu_1^{(j)}, \ldots, \mu_{s_j}^{(j)} \in \mathbb{C}_+$ of $\Lambda^{(j)}$. Then the product of the corresponding stability functions is given by

$$\prod_{j=1}^N R_{(j)}(z) = \prod_{j=1}^N \prod_{l=1}^{s_j} \frac{1 + \overline{\mu_l^{(j)}} z}{1 - \mu_l^{(j)} z}.$$

The same result can also be obtained with the one-dimensional tableaus $\Lambda_l^{(j)} = \mu_l^{(j)}$ and $\beta_l^{(j)} = 2\operatorname{Re}(\mu_l^{(j)})$ for $j = 1, \ldots, N$ and $l = 1, \ldots, s_j$, which also satisfy (16.5) and so yield the same approximate solution. Thus, to produce iterates which lie on the rank-one residual manifold, it is sufficient to use one-dimensional tableaus. A variant

Algorithm 16.1 Gramian quadrature algorithm on rank-one residual manifold via 1-stage Runge-Kutta methods

Input: $A \in \mathbb{R}^{n\times n}$ stable, $B \in \mathbb{R}^{n\times 1}$, parameters $\{\mu_1, \dots, \mu_N\} \subset \mathbb{C}_+$
Output: $Z \in \mathbb{C}^{n\times sN}$ with $ZZ^* \approx \mathcal{P}$, residual factor h_N

1: initialize $h_0 = B$, $Z_0 = [\,]$
2: **for** $j = 1, \dots, N$ **do**
3: solve $(I_n - \mu_j A)\mathcal{H}_j = h_{j-1}$ for $\mathcal{H}_j \in \mathbb{C}^{n\times 1}$
4: update $Z_j = [Z_{j-1}, \mathcal{H}_j\sqrt{2\,\mathrm{Re}(\mu_j)}]$
5: $h_j = h_{j-1} + 2\,\mathrm{Re}(\mu_j)A\mathcal{H}_j$
6: **end for**
7: $Z = Z_N$

of Algorithm 15.1 tailored for one-dimensional tableaus which satisfy (16.5) is given in Algorithm 16.1.

Remark 16.2. The approximate Cholesky factor Z from Algorithm 16.1 spans a rational Krylov subspace with poles given by μ_j^{-1} due to Theorem 15.3 and the Lyapunov residual is of rank one due to Theorem 16.1. Thus the iterates produced in Algorithm 16.1 are ADI iterates. Further, the ADI iterates are unique as shown in Theorem 8.3, so Algorithm 16.1 is equivalent to Algorithm 4.1 with $\alpha_j = -\mu_j^{-1}$. For an explicit proof of the equivalence of both algorithms we refer to our paper [21, Thm. 3].

Note that in general larger tableaus allow for more accurate quadrature rules. This is not the case for tableaus which satisfy (16.5), i.e. with iterates on the rank-one residual manifold, as such s-stage Runge-Kutta methods can be reduced to a series of 1-stage Runge-Kutta methods as described above.

For a good approximation the choice of the parameters $\mu_1, \dots, \mu_N$ in Algorithm 16.1 is crucial. We know that the Lyapunov residual for iterates which fulfill the invariant (16.2) is given by $h_N h_N^*$. Hence for a small residual the norm $\|h_N\|$ has to be small. Consider Algorithm 16.1, that is, $R_{(j)}(z) = \frac{1+\overline{\mu_j}z}{1-\mu_j z}$. Then with the iteration rule (16.8)

and the iteration matrix (16.10) we find

$$\begin{aligned} h_N &= \left(\prod_{j=1}^{N} M_j\right) B \\ &= V \left(\prod_{j=1}^{N} \begin{bmatrix} R_{(j)}(\lambda_1) & & \\ & \ddots & \\ & & R_{(j)}(\lambda_n) \end{bmatrix}\right) V^{-1} B \\ &= V \left(\prod_{j=1}^{N} \begin{bmatrix} \frac{1+\overline{\mu_j}\lambda_1}{1-\mu_j\lambda_1} & & \\ & \ddots & \\ & & \frac{1+\overline{\mu_j}\lambda_n}{1-\mu_j\lambda_n} \end{bmatrix}\right) V^{-1} B. \end{aligned} \tag{16.12}$$

Taking the 2-norm implies

$$\|h_N\| \leq \|V\| \max_{i=1,\ldots,n} \left(\prod_{j=1}^{N} \frac{|1+\overline{\mu_j}\lambda_i|}{|1-\mu_j\lambda_i|}\right) \|V^{-1}\| \|B\|.$$

To minimize this error bound the parameters μ_j have to be chosen such that the problem

$$\min_{\mu_1,\ldots,\mu_N} \max_{i=1,\ldots,n} \prod_{j=1}^{N} \frac{|1+\overline{\mu_j}\lambda_i|}{|1-\mu_j\lambda_i|}$$

is solved. It is equivalent to the rational min-max problem [12, Sec. 2.2]. For $N = n$ the choice $\mu_j = -\overline{\lambda_j}^{-1}$, $j = 1, \ldots, N$ yields $h_N = 0$ and thus $P_N = \mathcal{P}$. However, in that case the final iterate Z_N of Algorithm 16.1 is of size $n \times n$. Thus, this would not be a low rank approximation. Still this suggests that the parameters should somehow approximate the negative conjugated inverse of the eigenvalues λ_i of A, i.e. $\mu_j \approx -\overline{\lambda_i}^{-1}$. As we consider a stable system matrix A, all eigenvalues λ_i have negative real part. Therefore, to obtain a factor with modulus smaller one, i.e. $\frac{|1+\overline{\mu_j}\lambda_i|}{|1-\mu_j\lambda_i|} < 1$, the parameters μ_j must have positive real parts. We note that in practice precomputed shifts as described above are outperformed by dynamic shift strategies, see e.g. [48, Ch. 5] and the references therein.

16.3. Realification

When the set of shifts in Algorithm 16.1 consists of complex conjugated pairs we can avoid complex arithmetic forced by the use of complex parameters μ_j. We use the fact that instead of two steps with the 1-stage tableaus $\Lambda^{(j)} = \mu$, $\beta^{(j)} = 2\operatorname{Re}(\mu)$ and $\Lambda^{(j+1)} = \overline{\mu}$, $\beta^{(j+1)} = 2\operatorname{Re}(\mu)$ (as in Algorithm 16.1) we can equivalently perform one step of Algorithm 15.1 with a suitable real 2-stage Butcher tableau. Then realification is applied as discussed in Section 15.2. Due to the special structure of the tableaus we can explicitly state the used similarity transformations here. The resulting procedure is similar to the realification approach described in Section 4 for the ADI iteration.

A suitable tableau is given by

$$\Lambda = \begin{bmatrix} \operatorname{Re}(\mu) & \operatorname{Re}(\mu) + |\mu| \\ \operatorname{Re}(\mu) - |\mu| & \operatorname{Re}(\mu) \end{bmatrix} \in \mathbb{R}^{2\times 2}, \ \beta = \begin{bmatrix} 2\operatorname{Re}(\mu) \\ 2\operatorname{Re}(\mu) \end{bmatrix}$$

as $\lambda(\Lambda) = \{\mu, \overline{\mu}\}$ holds and the tableau condition (16.5) is satisfied. One easily verifies

$$T\Lambda^{\mathsf{T}}T^{-1} = \begin{bmatrix} \operatorname{Re}(\mu) & \operatorname{Im}(\mu) \\ -\operatorname{Im}(\mu) & \operatorname{Re}(\mu) \end{bmatrix}$$

with

$$T = \begin{bmatrix} 1 & 1 \\ \frac{\operatorname{Re}(\mu)+|\mu|}{\operatorname{Im}(\mu)} & \frac{\operatorname{Re}(\mu)-|\mu|}{\operatorname{Im}(\mu)} \end{bmatrix} = \underbrace{\sqrt{2}\begin{bmatrix} 1 & 0 \\ \frac{\operatorname{Re}(\mu)}{\operatorname{Im}(\mu)} & \frac{|\mu|}{\operatorname{Im}(\mu)} \end{bmatrix}}_{=L} \underbrace{\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}}_{=:Q}.$$

Please note that L from above is essentially the same as L in the realification of the ADI iteration, i.e. in (4.1) with $\frac{|\mu|}{\operatorname{Im}(\mu)} = \sqrt{\left(\frac{\operatorname{Re}(\mu)}{\operatorname{Im}(\mu)}\right)^2 + 1}$ (for $\operatorname{Im}(\mu) > 0$), and Q is an orthogonal matrix. We want to solve

$$\mathcal{H}_j = [h_{j-1},\, h_{j-1}] + A\mathcal{H}_j\Lambda^{\mathsf{T}}$$

as in (15.16) with $[z_1,\, z_2] = [h_{j-1},\, h_{j-1}]$ as in (15.17). We find $[y_1,\, y_2] = [h_1,\, h_2]T^{-1} =$

$[h_{j-1},\, 0]$, and so we obtain $\mathcal{H}_j = [v_1,\, v_2]T$, where $[v_1,\, v_2]$ is determined via

$$(I_n - \mu A)(v_1 + \imath v_2) = h_{j-1}$$

as in (15.20).

Summing up, the Gramian approximation calculated via two steps of Algorithm 16.1 with parameters μ and $\overline{\mu}$ can also be obtained as follows with a realified variant of Algorithm 15.1: First, solve $(I_n - \mu A)v = h_{j-1}$ for $v = v_1 + \imath v_2$, then set $Z_{j+1} = \left[Z_{j-1},\, \sqrt{2\,\mathrm{Re}(\mu)}[v_1,\, v_2]T\right]$ and $h_{j+1} = h_{j-1} + 2\,\mathrm{Re}(\mu)A\left(2v_1 + 2\frac{\mathrm{Re}(\mu)}{\mathrm{Im}(\mu)}v_2\right)$, where we have used $T\beta = 2\,\mathrm{Re}(\mu)\left[2,\, 2\frac{\mathrm{Re}(\mu)}{\mathrm{Im}(\mu)}\right]^{\mathsf{T}}$ in the last step.

16.4. Shifted Legendre polynomials

A seemingly different approach for the solution of Lyapunov equations is presented in [72]. The approximate Cholesky factor of the Gramian is obtained via an approximation of the impulse response $h(t) = e^{At}B$ by shifted Legendre polynomials. Here we show that the resulting Gramian approximation is the same as with one step of Algorithm 15.1, the Gramian quadrature algorithm, using the Gauß-Legendre tableau of size s. It is thus an iteration on the low rank residual manifold and so another variant of the ADI iteration.

The method with shifted Legendre polynomials from [72] works as follows. With a fixed scalar $s \in \mathbb{N}$, $D = \mathrm{diag}(1, \sqrt{3}, \ldots, \sqrt{2s-1})$ and $\omega \in \mathbb{R}_+$ the Gramian approximation is given by $\mathcal{P} \approx FF^{\mathsf{T}}$ with $F = \widetilde{F} \cdot D^{-\frac{1}{2}} \cdot \sqrt{\omega}$. The matrix $\widetilde{F} \in \mathbb{R}^{n\times s}$ is determined via the ns-dimensional block tridiagonal system of linear equations

$$(I_{ns} - \omega(\Lambda' \otimes A))\,\mathrm{vec}(\widetilde{F}) = B \otimes e_1 \tag{16.13}$$

with

$$\Lambda' = \begin{bmatrix} \frac{1}{2} & -\frac{1}{6} & & & \\ \frac{1}{2} & 0 & -\frac{1}{10} & & \\ & \frac{1}{6} & 0 & \ddots & \\ & & \ddots & \ddots & -\frac{1}{2(2s+1)} \\ & & & \frac{1}{2(2s-1)} & 0 \end{bmatrix}.$$

We now show equivalence of this approximation and the approximation generated by Algorithm 15.1. Let $\mathcal{H}' = \widetilde{F} \cdot D^{-1}$. De-vectorizing (16.13) yields

$$\mathcal{H}'D - \omega A \mathcal{H}' D (\Lambda')^{\mathsf{T}} = [B, 0, \ldots, 0]$$

and multiplication with D^{-1} from the right results in

$$\mathcal{H}' = [B, 0, \ldots, 0] + \omega A \mathcal{H}' (D^{-1}\Lambda' D)^{\mathsf{T}}. \tag{16.14}$$

We next show that $D^{-1}\Lambda' D = X_{\mathrm{G}}$ holds for the matrix

$$X_{\mathrm{G}} = \begin{bmatrix} \frac{1}{2} & -\xi_1 & & \\ \xi_1 & 0 & \ddots & \\ & \ddots & \ddots & -\xi_{s-1} \\ & & \xi_{s-1} & 0 \end{bmatrix}$$

from [25, Lem. 359A] with $\xi_i = \frac{1}{2\sqrt{4i^2-1}}$. Multiplication of Λ' with D^{-1} from the left means that the ith row is scaled with the factor $\frac{1}{\sqrt{2i-1}}$ while multiplication with D from the right means scaling of the ith column with $\sqrt{2i-1}$. Thus we find for the upper and

lower diagonal entries of $D^{-1}\Lambda' D$ that

$$(D^{-1}\Lambda' D)_{i,i+1} = \frac{\sqrt{2(i+1)-1}}{\sqrt{2i-1}} \cdot \frac{-1}{2(2i+1)} = \frac{-1}{2\sqrt{4i^2-1}},$$
$$(D^{-1}\Lambda' D)_{i+1,i} = \frac{\sqrt{2i-1}}{\sqrt{2(i+1)-1}} \cdot \frac{1}{2(2i-1)} = \frac{1}{2\sqrt{4i^2-1}}$$

holds for $i = 1, \ldots, s-1$, which proves $D^{-1}\Lambda' D = X_{\mathrm{G}}$.

Let Λ, β and γ be as in the Butcher tableau for the Gauß-Legendre method of size s. Define the generalized Vandermonde matrix as in [25, Sec. 359]

$$W \coloneqq \begin{bmatrix} P_0(\gamma_1) & \cdots & P_{s-1}(\gamma_1) \\ \vdots & \ddots & \vdots \\ P_0(\gamma_s) & \ldots & P_{s-1}(\gamma_s) \end{bmatrix}$$

with the normalized Legendre polynomials P_k of degree k for $k = 0, \ldots, s-1$ on the interval $[0, 1]$ so that

$$\sum_{i=1}^{s} \beta_i P_k(c_i) P_l(c_i) = \delta_{kl}$$

holds. This implies orthonormality of the matrix $\operatorname{diag}(\beta)^{\frac{1}{2}} W$ as well as the equations

$$W^{\mathsf{T}} = (\operatorname{diag}(\beta) W)^{-1},$$
$$e_1^{\mathsf{T}} = \mathbb{1}_s^{\mathsf{T}} \operatorname{diag}(\beta) W,$$

so that

$$\begin{aligned} [B, 0, \ldots, 0] &= e_1^{\mathsf{T}} \otimes B \\ &= (\mathbb{1}_s^{\mathsf{T}} \cdot \operatorname{diag}(\beta) W) \otimes (B \cdot 1) \\ &= (\mathbb{1}_s^{\mathsf{T}} \otimes B) \cdot (\operatorname{diag}(\beta) W \otimes 1) \\ &= [B, \ldots, B] \operatorname{diag}(\beta) W \end{aligned}$$

holds. Further, due to [25, Lem. 359A], we have

$$X_{\mathrm{G}} = W^{\mathsf{T}} \operatorname{diag}(\beta)\Lambda W.$$

Herewith (16.14) becomes

$$\mathcal{H}' = [B, \dots, B] \operatorname{diag}(\beta)W + \omega A \mathcal{H}' W^{\mathsf{T}} \Lambda^{\mathsf{T}} \operatorname{diag}(\beta) W. \tag{16.15}$$

Multiplication of (16.15) with $W^{\mathsf{T}} = (\operatorname{diag}(\beta)W)^{-1}$ from the right and setting $\mathcal{H} = \mathcal{H}'W^{\mathsf{T}}$ yields

$$\mathcal{H} = [B, \dots, B] + \omega A \mathcal{H} \Lambda^{\mathsf{T}}.$$

Due to line 4 in Algorithm 15.1 the approximate Cholesky factor using the Gauß-Legendre method is given by $Z = \mathcal{H} \operatorname{diag}(\omega\beta)^{\frac{1}{2}}$. Therefore, because of

$$\begin{aligned} ZZ^{\mathsf{T}} &= \mathcal{H} \operatorname{diag}(\omega\beta)\mathcal{H}^{\mathsf{T}} \\ &= \omega \mathcal{H}' \underbrace{W^{\mathsf{T}} \operatorname{diag}(\beta) W}_{=I_s} (\mathcal{H}')^{\mathsf{T}} \\ &= \omega \widetilde{F} D^{-\frac{1}{2}} (\widetilde{F} D^{-\frac{1}{2}})^{\mathsf{T}} \\ &= FF^{\mathsf{T}}, \end{aligned}$$

the approximation FF^{T} based on Legendre polynomials and the approximation ZZ^{T} based on the Gauß-Legendre method is the same. An efficient way to compute the approximation is given by Algorithm 16.1 with parameters $\mu_1, \dots, \mu_s$ chosen as the eigenvalues of the matrix ωX_{G} from (16.14), which is similar to $\omega\Lambda$ from the Gauß-Legendre tableau.

16.5. Generalized Lyapunov equations and residual norm evaluation

To handle generalized Lyapunov equations

$$A\mathcal{P}E^{\mathsf{T}} + E\mathcal{P}A^{\mathsf{T}} + BB^{\mathsf{T}} = 0$$

with $B \in \mathbb{R}^{n\times m}$ efficiently Algorithm 15.1 has to be adapted as follows.

To treat equations with the system matrix E, line 3 of Algorithm 15.1 is modified to

$$E\mathcal{H}_j = [h_{j-1}, \ldots, h_{j-1}] + \omega_j A\mathcal{H}_j\Lambda^{\mathsf{T}}. \tag{16.16}$$

To solve for $\mathcal{H}_j$ efficiently the method described in Section 15.2 has to be adapted accordingly. We add E on the left-hand side of (15.14) and obtain

$$E\mathcal{H}'_j = [\alpha_1 h_{j-1}, \ldots, \alpha_s h_{j-1}] + \omega_j A\mathcal{H}'_j(\Lambda')^{\mathsf{T}}. \tag{16.17}$$

Further, (15.15) becomes

$$(E - \omega_j\mu_i A)\mathfrak{h}_i'^{(j)} = \alpha_i h_{j-1} + \omega_j \sum_{l=1}^{i-1} \lambda'_{il} A\mathfrak{h}_l'^{(j)}.$$

When $B \in \mathbb{R}^{n\times m}$ has multiple columns, Λ and Λ' in (16.16) and (16.17) have to be replaced by $\Lambda \otimes I_m$ and $\Lambda' \otimes I_m$. Line 5 of Algorithm 15.1 becomes $h_j = h_{j-1} + \omega_j A\mathcal{H}_j(\beta \otimes I_m)$ with $h_j \in \mathbb{C}^{n\times m}$. Further the relation $\mathcal{H}_j = \mathcal{H}'_j(S \otimes I_m)$ has to be used.

The residual $\mathcal{L}(Z_jZ_j^*)$ is of size $n \times n$ and in practice n is large. For a fast calculation of the residual norm $\|\mathcal{L}(Z_jZ_j^*)\|_{\mathrm{F}}$ we make use of (16.4), which can be rewritten as follows

$$\mathcal{L}(ZZ^*) = \underbrace{[h_N, K_1, \ldots, K_N]}_{=:V} \underbrace{\left(\begin{bmatrix} 1 & \\ & \operatorname{diag}(\omega_1^2, \ldots, \omega_N^2) \otimes M \end{bmatrix} \otimes I_m \right)}_{=:M_0} [h_N, K_1, \ldots, K_N]^*,$$

with the Hermitian matrix $M = \operatorname{diag}(\beta)\overline{\Lambda} + \Lambda^{\mathsf{T}} \operatorname{diag}(\beta) - \beta\beta^{\mathsf{T}}$. Recall that Z is a low

rank approximation of dimension $n \times (1+Ns)m$ with $(1+Ns)m \ll n$. Thus M_0 is a small matrix of size $(1+Ns)m$. To evaluate the residual norm we use

$$\begin{aligned}\|\mathcal{L}(ZZ^*)\|_{\mathrm{F}} = \|VM_0V^*\|_{\mathrm{F}} &= \operatorname{trace}(\underbrace{VM_0V^*VM_0V^*}_{n\times n})^{\frac{1}{2}} \\ &= \operatorname{trace}(\underbrace{M_0V^*VM_0V^*V}_{(1+Ns)m\times(1+Ns)m})^{\frac{1}{2}}, \end{aligned} \tag{16.18}$$

which holds because of the invariance of the trace under cyclic permutations of its argument. The trace of the small matrix in (16.18) can be calculated rapidly. The slowest part is the calculation of V^*V. As V is constructed column by column only the scalar products with the new entries should be calculated and stored in every step of the iteration. If $M = 0$ holds, then the residual norm evaluation simplifies considerably to

$$\|\mathcal{L}(ZZ^*)\|_{\mathrm{F}} = \|h_N h_N^*\|_{\mathrm{F}} = \|h_N^* h_N\|_{\mathrm{F}}.$$

16.6. Numerical experiments

In this section numerical experiments are presented to illustrate some properties of the geometric integration method introduced in this section. For the experiments the implementation of Algorithm 15.1 was modified as described above such that generalized Lyapunov equations can be handled.

The two examples used here are chip0 and rail79k from the Oberwolfach model reduction benchmark collection as introduced in Section 2.6. All numerical experiments were executed using MATLAB 2019a on an Intel® Core™ i7-5600U CPU @ 2.60 GHz with 12 GB RAM.

In order to assess different Runge-Kutta methods, the change of the Lyapunov residual $\mathcal{L}(Z_jZ_j^*)$ and the iterate h_j is examined. Both quantities are coupled through (16.4) and a small residual indicates a good approximate solution. The relative Lyapunov residual norm $\|\mathcal{L}(Z_jZ_j^*)\|_{\mathrm{F}}/\|BB^{\mathsf{T}}\|_{\mathrm{F}}$ and the relative norm $\|h_j\|_{\mathrm{F}}/\|B\|_{\mathrm{F}}$ are determined after each iteration step.

The first method used in the experiments is the Radau IA method [25, Sec. 344] with the Butcher tableau

$$\begin{array}{c|ccc} 0 & \frac{1}{9} & \frac{-1-\sqrt{6}}{18} & \frac{-1+\sqrt{6}}{18} \\ \frac{3}{5}-\frac{\sqrt{6}}{10} & \frac{1}{9} & \frac{11}{45}+\frac{7\sqrt{6}}{360} & \frac{11}{45}-\frac{43\sqrt{6}}{360} \\ \frac{3}{5}+\frac{\sqrt{6}}{10} & \frac{1}{9} & \frac{11}{45}+\frac{43\sqrt{6}}{360} & \frac{11}{45}-\frac{7\sqrt{6}}{360} \\ \hline & \frac{1}{9} & \frac{4}{9}+\frac{\sqrt{6}}{36} & \frac{4}{9}-\frac{\sqrt{6}}{36} \end{array} \tag{16.19}$$

of size 3, denoted by *Radau IA (3)*. It was chosen as Λ has only nonzero eigenvalues, no entry in β is zero and the iterates do not lie on the manifold $\mathcal{M}$ from (16.3), because the tableau condition (16.5) is not satisfied. Other methods like Radau IIA and Lobatto IIIC were also tested and showed a similar behavior as the Radau IA method. The second method used is a DIRK method (16.7) where the diagonal elements μ_1, μ_2, μ_3 are chosen to be the eigenvalues of Λ from (16.19), so, in contrast to the Radau IA method, the iterates lie on the manifold $\mathcal{M}$. This method is denoted by $\mathcal{M}$*-Radau IA (3)*. Further, the Gauß-Legendre method with Butcher tableau

$$\begin{array}{c|ccc} \frac{1}{2}-\frac{\sqrt{15}}{10} & \frac{5}{36} & \frac{2}{9}-\frac{\sqrt{15}}{15} & \frac{5}{36}-\frac{\sqrt{15}}{30} \\ \frac{1}{2} & \frac{5}{36}+\frac{\sqrt{15}}{24} & \frac{2}{9} & \frac{5}{36}-\frac{\sqrt{15}}{24} \\ \frac{1}{2}+\frac{\sqrt{15}}{10} & \frac{5}{36}+\frac{\sqrt{15}}{30} & \frac{2}{9}+\frac{\sqrt{15}}{15} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \end{array}$$

is used, denoted by *Gauß-Leg. (3)*. The iterates lie on the manifold $\mathcal{M}$ as the tableau condition (16.5) is satisfied. We note that due to different eigenvalues of Λ in *Gauß-Leg. (3)* and $\mathcal{M}$*-Radau IA (3)* the methods are not equivalent.

In Figure 16.2 the relative norm of the Lyapunov residual and the relative norm of the iterate h_j is displayed for the chip0 example. As this example has single input and the Butcher tableaus are of size 3, after 20 steps of Algorithm 15.1 the approximate Cholesky factor Z consists of 60 columns. On the left-hand side the convergence of the methods is displayed. It is comparable for the methods $\mathcal{M}$*-Radau IA (3)* and *Gauß-Leg. (3)* and stagnates after 48 columns at 10^{-9}, while the residual obtained through the

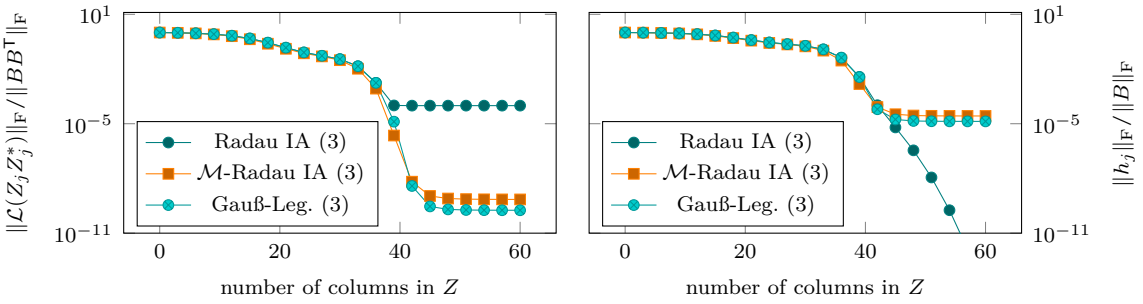

Figure 16.2.: The chip0 example with dimensions $n = 20082$, $m = 1$ and time step sizes `logspace(-5,2,20)`. Relative norms of the residual (left) and relative norms of the iterates h_j (right).

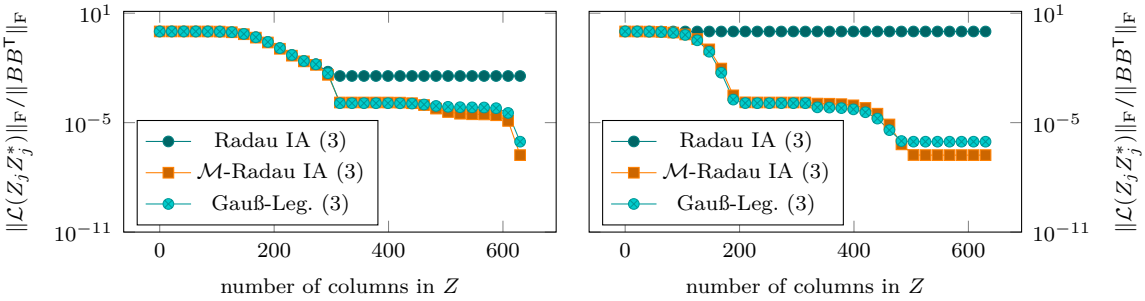

Figure 16.3.: Relative residual norms for the rail79k example with dimensions $n = 79841$, $m = 7$. Time step sizes `[logspace(-6,5,15), logspace(-6,5,15)]` (left) and `[logspace(5,-6,15), logspace(5,-6,15)]` (right).

Radau IA (3) method stagnates after 39 columns at 10^{-4}. On the right-hand side the relative norm of the iterate h_j is displayed. While for the two methods with iterates on $\mathcal{M}$ the relative norm of h_j correlates with the relative residual norm on the left-hand side in accordance with $\|\mathcal{L}(Z_jZ_j^*)\|_F = \|h_jh_j^*\|_F = \|h_j\|_F^2$, this is not the case for the method *Radau IA (3)*. This behavior is in agreement with the invariance equation (16.2), which for iterates on $\mathcal{M}$ connects the Lyapunov residual and $h_jh_j^*$. On the other hand, for iterates not lying on $\mathcal{M}$, we have to apply the more general sum (16.4). That is, the Lyapunov residual consists of $h_jh_j^*$ and terms depending on $K_i = A\mathcal{H}_i$ for $i = 1, \ldots, j$.

The convergence plots for the rail79k example are displayed in Figure 16.3. With 7 inputs and Butcher tableaus of size 3, after 30 steps of Algorithm 15.1 the approximate Cholesky factor Z is made up of 630 columns. We investigated the sensitivity of the methods to the choice of the time step sizes. Therefore 15 steps with increasing (de-

creasing) time step sizes on the left-hand (right-hand) side were performed, then the same 15 time step sizes were used again. For the increasing time step sizes on the left-hand side the first half of the residual norm plot shows the same behavior as for the previous example. Using the identical 15 time step sizes a second time, it can be seen that the residual norms for the two methods with iterates on $\mathcal{M}$ decrease further, while for the *Radau IA (3)* method it stagnates at the same level as after the first 15 steps. On the right-hand side we observe that starting with larger time step sizes is advantageous for the methods with iterates on $\mathcal{M}$, as the decay of the residual norm is faster at the beginning (but results in the same final residual due to (16.12)). For the Radau IA (3) method however we do not observe any convergence.

The first numerical experiment shows the effects of Theorem 16.1. In the methods which satisfy the tableau condition (16.5), a direct connection between $\mathcal{L}(Z_j Z_j^*)$ and h_j exists, while for other methods the K_i from the sum (16.4) are relevant, too. The second experiment was used to demonstrate that the methods with iterates on $\mathcal{M}$ are robust regarding the order choice of the time step sizes. That is, the order of the time step sizes does not alter the final approximation, in agreement with (16.12). On the other hand, in methods whose iterates do not lie on $\mathcal{M}$ no convergence takes place if too large time step sizes are chosen.

Our experiments indicate that those Runge-Kutta methods with iterates on $\mathcal{M}$ (i.e. the same iterates as generated in the ADI iteration) have properties which are advantageous over other Runge-Kutta methods for the approximate solution of the Lyapunov equation.

17. Sylvester equation

In this section we consider the Sylvester equation

$$A\mathcal{Y} - \mathcal{Y}B - FG^{\mathsf{T}} = 0 \qquad \text{(2.2 revisited)}$$

with the system matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $F \in \mathbb{R}^{n\times r}$ and $G \in \mathbb{R}^{m\times r}$, $r \leq n, m$. We assume that the spectra of A and B are disjoint, $\lambda(A) \cap \lambda(B) = \emptyset$, as then (2.2) has a unique solution.

Similar to the Lyapunov case where we only considered Lyapunov equations $A\mathcal{P} + \mathcal{P}A^{\mathsf{T}} = -BB^{\mathsf{T}}$ with rank-one right-hand side, we will restrict our discussion to the case $r = 1$, $F = f$, $G = g$ here. The case $r > 1$ with $F = [f_1, \ldots, f_r]$ and $G = [g_1, \ldots, g_r]$ can be reduced to $\mathcal{Y} = \sum_{i=1}^{r} \mathcal{Y}_i$ with $A\mathcal{Y}_i - \mathcal{Y}_i B = f_i g_i^{\mathsf{T}}, i = 1, \ldots, r$.

In analogy to (15.1) and (15.2) we consider the system of ODEs

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} Y(t) &= \hat{h}(t)\check{h}(t)^{\mathsf{T}}, & Y(0) &= 0, \\ \frac{\mathrm{d}}{\mathrm{d}t} \hat{h}(t) &= A\hat{h}(t), & \hat{h}(0) &= f, \\ \frac{\mathrm{d}}{\mathrm{d}t} \check{h}(t) &= -B^{\mathsf{T}}\check{h}(t), & \check{h}(0) &= -g. \end{aligned} \tag{17.1}$$

Please note that for $t \to \infty$ in general we will have $Y(t) \not\to \mathcal{Y}$. Nonetheless, as we will see, (17.1) is useful in order to derive approximations to $\mathcal{Y}$.

Due to the product rule the second derivative of Y satisfies

$$\begin{aligned} \frac{\mathrm{d}^2}{\mathrm{d}t^2} Y(t) &= \frac{\mathrm{d}}{\mathrm{d}t} \hat{h}(t)\check{h}(t)^{\mathsf{T}} \\ &= A\hat{h}(t)\check{h}(t)^{\mathsf{T}} - \hat{h}(t)\check{h}(t)^{\mathsf{T}}B \\ &= A\left(\frac{\mathrm{d}}{\mathrm{d}t} Y(t)\right) - \left(\frac{\mathrm{d}}{\mathrm{d}t} Y(t)\right) B. \end{aligned}$$

By integrating both sides over the interval $[0, t]$ we find

$$\frac{\mathrm{d}}{\mathrm{d}t} Y(t) = AY(t) - Y(t)B - fg^{\mathsf{T}}.$$

Thus the solution of the system of ODEs (17.1) does not satisfy (2.2) exactly, a rank one residual does remain

$$AY(t) - Y(t)B - fg^{\mathsf{T}} = \hat{h}(t)\check{h}(t)^{\mathsf{T}}.$$

We therefore intend to apply numerical quadrature methods to (17.1) which preserve the rank-one residual.

17.1. Approximating $\mathcal{Y}$ by Runge-Kutta methods

We propose to solve the partitioned system of ODEs (17.1) in two main steps, in analogy to the approach for the Lyapunov equation in Section 15. First, the latter two equations for $\hat{h}(t)$ and $\check{h}(t)$ are solved, then the equation for $Y(t)$ is solved. To solve the three ODEs we will make use of three different s-stage Runge-Kutta methods: The function $Y(t)$ is approximated using the Butcher tableau with $\Lambda^{(j)} \in \mathbb{C}^{s\times s}$ and $\beta^{(j)} \in \mathbb{C}^s$, the function $\hat{h}(t)$ with $\hat{\Lambda}^{(j)} \in \mathbb{C}^{s\times s}$ and $\hat{\beta}^{(j)} \in \mathbb{C}^s$ and the function $\check{h}(t)$ with $\check{\Lambda}^{(j)} \in \mathbb{C}^{s\times s}$ and $\check{\beta}^{(j)} \in \mathbb{C}^s$. As before, the upper index j is used to denote the jth step of the iteration and without loss of generality the time step sizes are all set to $\omega_j = 1$.

In analogy to the derivation in the Lyapunov case we find that the application of the Runge-Kutta methods for $\hat{h}(t)$ and $\check{h}(t)$ lead to the iterations

$$\hat{h}_j = \hat{h}_{j-1} + \hat{K}_j\hat{\beta}^{(j)}, \tag{17.2}$$

$$\check{h}_j = \check{h}_{j-1} + \check{K}_j\check{\beta}^{(j)}, \tag{17.3}$$

with $\hat{h}_0 = f$, $\check{h}_0 = -g$ (see (15.9)), $\hat{K}_j = A\hat{\mathcal{H}}_j$ and additionally $\check{K}_j = -B^{\mathsf{T}}\check{\mathcal{H}}_j$ (see (15.7)). Further

$$\hat{\mathcal{H}}_j = \left[\hat{h}_{j-1}, \ldots, \hat{h}_{j-1}\right] + A\hat{\mathcal{H}}_j(\hat{\Lambda}^{(j)})^{\mathsf{T}}, \tag{17.4}$$

$$\check{\mathcal{H}}_j = \left[\check{h}_{j-1}, \ldots, \check{h}_{j-1}\right] - B^{\mathsf{T}}\check{\mathcal{H}}_j(\check{\Lambda}^{(j)})^{\mathsf{T}}, \tag{17.5}$$

holds (see (15.8)). The solution of (17.4) is unique if and only if

$$\hat{\mu}_p \neq \hat{\lambda}_q^{-1},$$

for all $\hat{\mu}_p \in \lambda(\hat{\Lambda}^{(j)})$ and all $\hat{\lambda}_q \in \lambda(A)$, while the solution of (17.5) is unique if and only

if

$$\breve{\mu}_k \neq -\breve{\lambda}_\ell^{-1}$$

for all $\breve{\mu}_k \in \lambda(\breve{\Lambda}^{(j)})$ and all $\breve{\lambda}_\ell \in \lambda(B)$ (see (15.11)).

The jth iterate Y_j approximating the function $Y(t)$ is then given by

$$Y_j = Y_{j-1} + \hat{\mathcal{H}}_j \operatorname{diag}(\beta^{(j)}) \breve{\mathcal{H}}_j^*$$

where $Y_0 = 0$ (see the derivations leading to (15.9), in particular note that Λ does not appear, as the right-hand side of $Y'(t) = \hat{h}(t)\breve{h}(t)^\mathsf{T}$ is independent of $Y(t)$). We rewrite Y_j in terms of two low rank factors and a diagonal matrix

$$Y_j = \hat{Z}_j \Gamma_j \breve{Z}_j^*$$

with

$$\hat{Z}_j = \left[\hat{Z}_{j-1}, \hat{\mathcal{H}}_j\right], \quad \breve{Z}_j = \left[\breve{Z}_{j-1}, \breve{\mathcal{H}}_j\right], \quad \Gamma_j = \operatorname{diag}(\Gamma_{j-1}, \beta^{(j)}).$$

The solution of the Sylvester equation is neither symmetric nor positive definite, so the factors $\hat{Z}_j$ and $\breve{Z}_j$ need not be equal and Γ_j may contain arbitrary complex valued entries.

Employing our main idea from Section 16 we only consider Runge-Kutta methods whose iterates preserve the low rank property of the Sylvester residual, that is

$$AY_j - Y_j B - fg^\mathsf{T} = \hat{h}_j \breve{h}_j^\mathsf{T}. \tag{17.6}$$

Thus the iterates Y_j will not necessarily approximate the function $Y(t)$ very well, but Y_j is a good approximation to $\mathcal{Y}$ when the residual $\hat{h}_j \breve{h}_j^\mathsf{T}$ is small.

17.2. Runge-Kutta methods preserving an invariant

To characterize the tableaus which ensure that all iterates fulfill the rank-one residual condition, we insert the first iterates $Y_1, \hat{h}_1$ and $\check{h}_1$ into (17.6). For the left-hand side we obtain

$$\begin{aligned}
AY_1 - Y_1B - fg^{\mathsf{T}} &= A(Y_0 + \hat{\mathcal{H}}_1 \operatorname{diag}(\beta^{(1)})\check{\mathcal{H}}_1^*) - (Y_0 + \hat{\mathcal{H}}_1 \operatorname{diag}(\beta^{(1)})\check{\mathcal{H}}_1^*)B - fg^{\mathsf{T}} \\
&= A\hat{\mathcal{H}}_1 \operatorname{diag}(\beta^{(1)})\check{\mathcal{H}}_1^* - \hat{\mathcal{H}}_1 \operatorname{diag}(\beta^{(1)})\check{\mathcal{H}}_1^* B - fg^{\mathsf{T}} \\
&= \hat{K}_1 \operatorname{diag}(\beta^{(1)})\check{\mathcal{H}}_1^* + \hat{\mathcal{H}}_1 \operatorname{diag}(\beta^{(1)})\check{K}_1^* - fg^{\mathsf{T}} \\
&= \hat{K}_1 \operatorname{diag}(\beta^{(1)}) \left([\check{h}_0, \ldots, \check{h}_0] + \check{K}_1(\check{\Lambda}^{(1)})^{\mathsf{T}}\right)^* \\
&\quad + \left([\hat{h}_0, \ldots, \hat{h}_0] + \hat{K}_1(\hat{\Lambda}^{(1)})^{\mathsf{T}}\right) \operatorname{diag}(\beta^{(1)})\check{K}_1^* - fg^{\mathsf{T}} \\
&= \hat{K}_1(\beta^{(1)})\check{h}_0^* + \hat{K}_1 \operatorname{diag}(\beta^{(1)})\overline{\check{\Lambda}^{(1)}}\check{K}_1^* \\
&\quad + \hat{h}_0(\beta^{(1)})^{\mathsf{T}}\check{K}_1^* + \hat{K}_1(\hat{\Lambda}^{(1)})^{\mathsf{T}} \operatorname{diag}(\beta^{(1)})\check{K}_1^* - fg^{\mathsf{T}},
\end{aligned}$$

while for the right-hand side

$$\begin{aligned}
\hat{h}_1\check{h}_1^* &= \left(\hat{h}_0 + \hat{K}_1\hat{\beta}^{(1)}\right)\left(\check{h}_0 + \check{K}_1\check{\beta}^{(1)}\right)^* \\
&= -fg^* + \hat{h}_0(\check{\beta}^{(1)})^*\check{K}_1^* \\
&\quad + \hat{K}_1\hat{\beta}^{(1)}\check{h}_0^* + \hat{K}_1\hat{\beta}^{(1)}(\check{\beta}^{(1)})^*\check{K}_1^*
\end{aligned}$$

holds. As g is real $g^{\mathsf{T}} = g^*$ and so the equation

$$\begin{aligned}
&\hat{K}_1\beta^{(1)}\check{h}_0^* + \hat{K}_1 \operatorname{diag}(\beta^{(1)})\overline{\check{\Lambda}^{(1)}}\check{K}_1^* + \hat{h}_0(\beta^{(1)})^{\mathsf{T}}\check{K}_1^* + \hat{K}_1(\hat{\Lambda}^{(1)})^{\mathsf{T}} \operatorname{diag}(\beta^{(1)})\check{K}_1^* \\
&\quad \stackrel{!}{=} \hat{h}_0(\check{\beta}^{(1)})^*\check{K}_1^* + \hat{K}_1\hat{\beta}^{(1)}\check{h}_0^* + \hat{K}_1\hat{\beta}^{(1)}(\check{\beta}^{(1)})^*\check{K}_1^*
\end{aligned}$$

has to hold in order to satisfy the invariance equation (17.6). This implies

$$\begin{aligned}
0 &= \hat{h}_0\left(\check{\beta}^{(1)} - \overline{\beta^{(1)}}\right)^*\check{K}_1^* + \hat{K}_1\left(\hat{\beta}^{(1)} - \beta^{(1)}\right)\check{h}_0^* \\
&\quad + \hat{K}_1\left(\hat{\beta}^{(1)}(\check{\beta}^{(1)})^* - \operatorname{diag}(\beta^{(1)})\overline{\check{\Lambda}^{(1)}} - (\hat{\Lambda}^{(1)})^{\mathsf{T}} \operatorname{diag}(\beta^{(1)})\right)\check{K}_1^*,
\end{aligned}$$

which is satisfied for $\hat{\beta}^{(1)} = \beta^{(1)}$, $\breve{\beta}^{(1)} = \overline{\beta^{(1)}}$, and

$$\operatorname{diag}(\beta^{(1)})\overline{\breve{\Lambda}^{(1)}} + (\hat{\Lambda}^{(1)})^{\mathsf{T}} \operatorname{diag}(\beta^{(1)}) - \beta^{(1)}(\beta^{(1)})^{\mathsf{T}} = 0.$$

Via induction this leads to the conditions

$$\begin{aligned} 0 &= \operatorname{diag}(\beta^{(j)})\overline{\breve{\Lambda}^{(j)}} + (\hat{\Lambda}^{(j)})^{\mathsf{T}} \operatorname{diag}(\beta^{(j)}) - \beta^{(j)}(\beta^{(j)})^{\mathsf{T}}, \\ \hat{\beta}^{(j)} &= \beta^{(j)}, \\ \breve{\beta}^{(j)} &= \overline{\beta^{(j)}}. \end{aligned} \tag{17.7}$$

Please note that $\breve{\Lambda}^{(j)}, \breve{\beta}^{(j)}$ and $\hat{\Lambda}^{(j)}, \hat{\beta}^{(j)}$ from the Butcher tableaus for approximating $\breve{h}(t)$ and $\hat{h}(t)$ are relevant here, as well as $\beta^{(j)}$ from the Butcher tableau for $Y(t)$, but as mentioned above not the matrix $\Lambda^{(j)}$. DIRK tableaus which fulfill (17.7) are given by

$$\hat{\Lambda} = \begin{bmatrix} \hat{\mu}_1 & 0 & \cdots & 0 \\ \beta_1 & \hat{\mu}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \beta_1 & \beta_2 & \cdots & \hat{\mu}_s \end{bmatrix}, \ \breve{\Lambda} = \begin{bmatrix} \breve{\mu}_1 & 0 & \cdots & 0 \\ \overline{\beta}_1 & \breve{\mu}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \overline{\beta}_1 & \overline{\beta}_2 & \cdots & \breve{\mu}_s \end{bmatrix}, \ \beta = \begin{bmatrix} \hat{\mu}_1 + \overline{\breve{\mu}_1} \\ \hat{\mu}_2 + \overline{\breve{\mu}_2} \\ \vdots \\ \hat{\mu}_s + \overline{\breve{\mu}_s} \end{bmatrix}. \tag{17.8}$$

17.3. Multiplicative update formulae for the residual factors $\hat{h}_j$ and $\breve{h}_j$

Let us assume that (17.7) holds with $\beta_k^{(j)} \neq 0$, $k = 1, \ldots, s$. Further assume that A and B are diagonalizable, that is, $A = \hat{V}\hat{D}\hat{V}^{-1}$ and $B = \breve{V}\breve{D}\breve{V}^{-1}$ with $\hat{D} = \operatorname{diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_n)$ and $\breve{D} = \operatorname{diag}(\breve{\lambda}_1, \ldots, \breve{\lambda}_m)$. Then, as in the case of the Lyapunov equation in Section 16.1, we find that the iterations (17.2) and (17.3) can be written in

multiplicative form as in (16.10)

$$\hat{h}_j = \hat{V} \begin{bmatrix} \hat{R}_j(\hat{\lambda}_1) & & & \\ & \hat{R}_j(\hat{\lambda}_2) & & \\ & & \ddots & \\ & & & \hat{R}_j(\hat{\lambda}_n) \end{bmatrix} \hat{V}^{-1} \hat{h}_{j-1}, \tag{17.9}$$

$$\check{h}_j = \check{V}^{-\mathsf{T}} \begin{bmatrix} \check{R}_j(-\check{\lambda}_1) & & & \\ & \check{R}_j(-\check{\lambda}_2) & & \\ & & \ddots & \\ & & & \check{R}_j(-\check{\lambda}_m) \end{bmatrix} \check{V}^{\mathsf{T}} \check{h}_{j-1}, \tag{17.10}$$

with the stability functions

$$\begin{aligned} \hat{R}_j(z) &= 1 + z(\beta^{(j)})^{\mathsf{T}} (I - z\hat{\Lambda}^{(j)})^{-1} \mathbb{1}_s, \\ \check{R}_j(z) &= 1 + z(\overline{\beta^{(j)}})^{\mathsf{T}} (I - z\check{\Lambda}^{(j)})^{-1} \mathbb{1}_s. \end{aligned}$$

As in the Lyapunov case, these stability functions only depend on the eigenvalues of the tableaus $\hat{\Lambda}^{(j)}$ and $\check{\Lambda}^{(j)}$. In order to see this, first observe that for $\beta_i^{(j)} \neq 0$, $i = 1, \ldots, s$, we find from (17.7) by multiplying with $\operatorname{diag}(\beta^{(j)})^{-1}$ from the right (respectively left)

$$\begin{aligned} 0 &= \operatorname{diag}(\beta^{(j)}) \overline{\check{\Lambda}^{(j)}} \operatorname{diag}(\beta^{(j)})^{-1} + (\hat{\Lambda}^{(j)})^{\mathsf{T}} - \beta^{(j)} \mathbb{1}_s^{\mathsf{T}}, \\ 0 &= \overline{\check{\Lambda}^{(j)}} + \operatorname{diag}(\beta^{(j)})^{-1} (\hat{\Lambda}^{(j)})^{\mathsf{T}} \operatorname{diag}(\beta^{(j)}) - \mathbb{1}_s (\beta^{(j)})^{\mathsf{T}}. \end{aligned}$$

As any matrix is similar to its transpose, these equations imply the similarities

$$\begin{aligned} \hat{\Lambda}^{(j)} - \mathbb{1}_s (\beta^{(j)})^{\mathsf{T}} &\sim -\overline{\check{\Lambda}^{(j)}}, \\ \check{\Lambda}^{(j)} - \mathbb{1}_s (\beta^{(j)})^{*} &\sim -\overline{\hat{\Lambda}^{(j)}}. \end{aligned} \tag{17.11}$$

Let $\hat{\Lambda}^{(j)}$ have eigenvalues $\hat{\mu}_1, \ldots, \hat{\mu}_s$ and let $\check{\Lambda}^{(j)}$ have eigenvalues $\check{\mu}_1, \ldots, \check{\mu}_s$. Now

Algorithm 17.1 Low rank solution to (2.2) via 1-stage Runge-Kutta methods

Input: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $f \in \mathbb{R}^{n\times 1}$, $g \in \mathbb{R}^{m\times 1}$, parameters $\{\hat{\mu}_1, \ldots, \hat{\mu}_N\} \subset \mathbb{C}$ and $\{\check{\mu}_1, \ldots, \check{\mu}_N\} \subset \mathbb{C}$

Output: matrices $\hat{Z} \in \mathbb{C}^{n\times N}$, $\check{Z} \in \mathbb{C}^{m\times N}$ and $\Gamma \in \mathbb{C}^{N\times N}$ with $\hat{Z}\Gamma\check{Z}^* \approx \mathcal{Y}$

initialize $\hat{h}_0 = f$, $\check{h}_0 = -g$, $\hat{Z}_0 = \check{Z}_0 = \Gamma_0 = [\,]$
for $j = 1, \ldots, N$ **do**
  solve $(I - \hat{\mu}_j A)\hat{\mathcal{H}}_j = \hat{h}_{j-1}$ for $\hat{\mathcal{H}}_j$
  solve $(I + \check{\mu}_j B^{\mathsf{T}})\check{\mathcal{H}}_j = \check{h}_{j-1}$ for $\check{\mathcal{H}}_j$
  update $\hat{Z}_j = [\hat{Z}_{j-1}, \hat{\mathcal{H}}_j]$
  update $\check{Z}_j = [\check{Z}_{j-1}, \check{\mathcal{H}}_j]$
  update $\Gamma_j = \operatorname{diag}(\Gamma_{j-1}, (\hat{\mu}_j + \overline{\check{\mu}}_j)I_r)$
  $\hat{h}_j = \hat{h}_{j-1} + (\hat{\mu}_j + \overline{\check{\mu}}_j)A\hat{\mathcal{H}}_j$
  $\check{h}_j = \check{h}_{j-1} - (\overline{\hat{\mu}}_j + \check{\mu}_j)B^{\mathsf{T}}\check{\mathcal{H}}_j$
end for
$\hat{Z} = \hat{Z}_N$, $\check{Z} = \check{Z}_N$, $\Gamma = \Gamma_N$

with (6.5) and (17.11) we have (similar to (16.11))

$$\begin{aligned}\hat{R}_j(z) &= \frac{\det(I - z(\hat{\Lambda}^{(j)} - \mathbb{1}_s(\beta^{(j)})^{\mathsf{T}}))}{\det(I - z\hat{\Lambda}^{(j)})} = \frac{\det(I + z\overline{\check{\Lambda}^{(j)}})}{\det(I - z\hat{\Lambda}^{(j)})} = \prod_{i=1}^{s} \frac{(1 + z\overline{\check{\mu}_i})}{(1 - z\hat{\mu}_i)},\\ \check{R}_j(z) &= \frac{\det(I - z(\check{\Lambda}^{(j)} - \mathbb{1}_s(\beta^{(j)})^{*}))}{\det(I - z\check{\Lambda}^{(j)})} = \frac{\det(I + z\overline{\hat{\Lambda}^{(j)}})}{\det(I - z\check{\Lambda}^{(j)})} = \prod_{i=1}^{s} \frac{(1 + z\overline{\hat{\mu}_i})}{(1 - z\check{\mu}_i)}.\end{aligned} \tag{17.12}$$

Thus, if the invariance condition (17.6) is enforced, then the stability functions of the Runge-Kutta method for $\hat{h}(t)$ and for $\check{h}(t)$ are not independent of each other. Both depend on the eigenvalues of $\hat{\Lambda}^{(j)}$ and $\check{\Lambda}^{(j)}$. Thus, also the iterates $\hat{h}_j$ (17.9) and $\check{h}_j$ (17.10) depend on those eigenvalues.

The stability functions corresponding to the s-stage DIRK tableaus as in (17.8) will have the form (17.12). As in Section 16.2 we can restrict ourselves to the use of several different 1-stage Butcher tableaus satisfying the tableau conditions (17.7), i.e. $\hat{\Lambda}^{(j)} = \hat{\mu}_j \in \mathbb{C}$ and $\check{\Lambda}^{(j)} = \check{\mu}_j \in \mathbb{C}$ with $\beta^{(j)} = \hat{\beta}^{(j)} = \hat{\mu}_j + \overline{\check{\mu}_j} \in \mathbb{C}$ and $\overline{\beta^{(j)}} = \check{\beta}^{(j)} = \overline{\hat{\mu}_j} + \check{\mu}_j \in \mathbb{C}$.

The resulting iteration for a low rank approximation to the solution of the Sylvester equation (2.2) is summarized in Algorithm 17.1. It is equivalent to [48, Alg. 3.4] with $\alpha_j = -\check{\mu}_j^{-1}$ and $\beta_j = \hat{\mu}_j^{-1}$ (and $E = I_n$, $C = I_m$, $r = 1$).

In the special case $B = -A^{\mathsf{T}}$ and $g = -f$ the Sylvester equation (2.2) reduces to the Lyapunov equation (1.2). If additionally $\hat{\mu}_j = \check{\mu}_j$, $j = 1, \ldots, N$, holds, then Algorithm 17.1 reduces to Algorithm 16.1 and we find $\hat{\mu}_j + \overline{\check{\mu}}_j = 2\,\mathrm{Re}(\hat{\mu}_j) \in \mathbb{R}$.

18. Conclusions

In this part we presented an approach to approximate linear matrix equations via numerical quadrature. The time-dependent Gramian was interpreted as the solution of a system of ODEs, which converges to the exact solution of the Lyapunov equation. To solve the occurring ODE an efficient way to apply Runge-Kutta methods was presented. The space spanned by the approximate Cholesky factor was identified to be a (rational) Krylov subspace. Hence, when the approximate Gramians are used for balancing related model order reduction, then the moments of the reduced system coincide with the moments of the original systems at the inverses of the (conjugated) eigenvalues of the Butcher tableaus multiplied with the time step sizes, while explicit quadrature methods correspond to interpolation at infinity.

A geometric integration approach was used to identify Runge-Kutta methods which preserve the rank of the initial residual. These methods were characterized via a simple condition on the corresponding Butcher tableaus. Stability functions of the Runge-Kutta methods were used to obtain an efficient residual based iteration which is equivalent to the ADI iteration. An approach for the Gramian approximation via Legendre polynomials was shown to be equivalent to our geometric integration approach when Gauß-Legendre methods are used. In numerical experiments the robustness of the geometrical integration approach was demonstrated.

All ideas are applied in slightly modified form to the Sylvester equation. Although the solution of the considered system of ODEs does not converge to the solution of the Sylvester equation, the application of Runge-Kutta methods which preserve the initial rank of the residual yields a sound approximation. That is, by retaining a geometric, qualitative property during the quadrature an ADI equivalent algorithm for the approximation of the solution of a Sylvester equation was derived from a system of ODEs.

A. Additional algorithms

Algorithm A.1 Lyapunov RADI iteration without RAD calculation (cf. Algorithm 9.2)

Input: system matrices A, B, C, set of shifts $s \subset \mathbb{C}$ with $s \cap -\overline{s} = \emptyset$
Output: approximate solution $Z_j Y_j Z_j^*$, residual factor R_j

initialize $Z_0 = [\,]$, $Y_0^{-1} = [\,]$, $R_0 = C^*$, $K_0 = 0$, $j = 0$
while not converged **do**
  obtain new shift(s) μ from s
  expand RAD, obtain $\tilde{Z}$, U_1, D ▷ solve $\tilde{Z} = (A^* - K_j B^* - \mu I_n)^{-1} R_j$
  solve $Y_{22} D + D^* Y_{22} - \tilde{Z}^* B B^* \tilde{Z} - U_1^* U_1 = 0$ for Y_{22}
  update $Z_{j+1} = [Z_j,\, \tilde{Z}]$, $Y_{j+1}^{-1} = \begin{bmatrix} Y_j^{-1} & 0 \\ 0 & Y_{22} \end{bmatrix}$
  update $R_{j+1} = R_j + \tilde{Z} Y_{22}^{-1} U_1^*$
  update $K_{j+1} = K_j + \tilde{Z} Y_{22}^{-1} (\tilde{Z}^* B)$
  $j = j + 1$
end while

Algorithm A.2 Riccati RAD iteration (R^2ADi) with E (see Section 10.1)

Input: system matrices A, E, B, C, set of shifts $s \subset \mathbb{C}$ with $s \cap -\overline{s} = \emptyset$
Output: approximate solution $Z_j Y_j Z_j^*$, residual factor R_j

initialize $Z_0 = [\,]$, $Y_0^{-1} = [\,]$, $R_0 = C^*$, $h_0 = [\,]$, $\underline{H_{-0}} = [\,]$, $s_0 = [\,]$, $j = 0$
while not converged **do**
  obtain new shift(s) μ from s
  expand RAD, obtain $\tilde{Z}$, U_1, D ▷ solve $\tilde{Z} = (A^* - \mu E^*)^{-1} R_j$
  solve $Y_{12} D + \underline{H_{-j}}^* Y_{12} - s_j^* (B^* \tilde{Z}) = 0$ for Y_{12}
  solve $Y_{12}^* Y_j h_j^* U_1 + Y_{22} D + U_1^* h_j Y_j Y_{12} + D^* Y_{22} - \tilde{Z}^* B B^* \tilde{Z} - U_1^* U_1 = 0$ for Y_{22}
  update $Z_{j+1} = [Z_j,\, \tilde{Z}]$, $Y_{j+1}^{-1} = \begin{bmatrix} Y_j^{-1} & Y_{12} \\ Y_{12}^* & Y_{22} \end{bmatrix}$
  update $R_{j+1} = C^* + E^* Z_{j+1} Y_{j+1} h_{j+1}^*$
  update $h_{j+1} = [h_j, U_1]$, $\underline{H_{-(j+1)}} = \left[\begin{array}{c|c} \underline{H_{-j}} & Y_j h_j^* U_1 \\ \hline 0 & D \end{array}\right]$
  update $s_{j+1} = [s_j,\, B^* \tilde{Z}]$
  $j = j + 1$
end while

Algorithm A.3 Lyapunov RADI iteration with E (see Section 10.1)

Input: system matrices A, E, B, C, set of shifts $s \subset \mathbb{C}$ with $s \cap -\overline{s} = \emptyset$
Output: approximate solution $Z_j Y_j Z_j^*$, residual factor R_j

initialize $Z_0 = [\,]$, $Y_0^{-1} = [\,]$, $R_0 = C^*$, $h_0 = [\,]$, $\underline{H_{-0}} = [\,]$, $s_0 = [\,]$, $K_0 = 0$, $j = 0$
while not converged **do**
  obtain new shift(s) μ from s
  expand RAD, obtain $\tilde{Z}$, U_1, D ▷ solve $\tilde{Z} = (A^* - K_j B^* - \mu E^*)^{-1} R_j$
  solve $Y_{22} D + D^* Y_{22} - \tilde{Z}^* B B^* \tilde{Z} - U_1^* U_1 = 0$ for Y_{22}
  update $Z_{j+1} = [Z_j,\, \tilde{Z}]$, $Y_{j+1}^{-1} = \begin{bmatrix} Y_j^{-1} & 0 \\ 0 & Y_{22} \end{bmatrix}$
  update $R_{j+1} = R_j + E^* \tilde{Z} Y_{22}^{-1} U_1^*$
  update $K_{j+1} = K_j + E^* \tilde{Z} Y_{22}^{-1} (\tilde{Z}^* B)$
  update $h_{j+1} = [h_j, U_1]$, $\underline{H_{-(j+1)}} = \begin{bmatrix} \underline{H_{-j}} & Y_j h_j^* U_1 + Y_j s_j^* (B^* \tilde{Z}) \\ 0 & D \end{bmatrix}$
  update $s_{j+1} = [s_j,\, B^* \tilde{Z}]$
  $j = j + 1$
end while

Algorithm A.4 Lyapunov ADI iteration (see Section 10.2)

Input: system matrices A, B, set of shifts $s \subset \mathbb{C}$ with $s \cap -\overline{s} = \emptyset$
Output: approximate solution $Z_j Y_j Z_j^*$, residual factor R_j

initialize $Z_0 = [\,]$, $Y_0^{-1} = [\,]$, $R_0 = B$, $j = 0$
while not converged **do**
  obtain new shift(s) μ from s
  expand RAD, obtain $\tilde{Z}$, U_1, D ▷ solve $\tilde{Z} = (A^* - \mu I_n)^{-1} R_j$
  solve $Y_{22} D + D^* Y_{22} - U_1^* U_1 = 0$ for Y_{22}
  update $Z_{j+1} = [Z_j,\, \tilde{Z}]$, $Y_{j+1}^{-1} = \begin{bmatrix} Y_j^{-1} & 0 \\ 0 & Y_{22} \end{bmatrix}$
  update $R_{j+1} = R_j + \tilde{Z} Y_{22}^{-1} U_1^*$
  $j = j + 1$
end while

Algorithm A.5 Lyapunov ADI iteration with simple RAD expansion (see Section 10.2)

Input: system matrices A, B, set of shifts $s \subset \mathbb{C}^+$
Output: approximate solution $Z_j Z_j^*$, residual factor R_j

initialize $Z_0 = [\,]$, $R_0 = B$, $j = 0$
while not converged **do**
  obtain new shift μ from s
  solve $\tilde{Z} = (A^* - \mu I_n)^{-1} R_j$
  update $Z_{j+1} = [Z_j,\, \sqrt{2\,\mathrm{Re}(\mu)}\,\tilde{Z}]$
  update $R_{j+1} = R_j + 2\,\mathrm{Re}(\mu)\,\tilde{Z}$
  $j = j + 1$
end while

References

[1] Mian I. Ahmad, Imad Jaimoukha, and Michalis Frangos. "Krylov subspace restart scheme for solving large-scale Sylvester equations". In: *Proceedings of the 2010 American Control Conference*. June 2010, pp. 5726–5731. DOI: 10.1109/ACC.2010.5531144.

[2] Luca Amodei and Jean-Marie Buchot. "An invariant subspace method for large-scale algebraic Riccati equation". In: *Applied Numerical Mathematics* 60.11 (2010). Special Issue: 9th IMACS International Symposium on Iterative Methods in Scientific Computing (IISIMSC 2008), pp. 1067–1082. DOI: 10.1016/j.apnum.2009.09.006.

[3] Athanasios C. Antoulas. *Approximation of Large-scale Dynamical Systems*. Society for Industrial and Applied Mathematics, 2005. ISBN: 978-0-89871-529-3. DOI: 10.1137/1.9780898718713.

[4] R. H. Bartels and G. W. Stewart. "Solution of the Matrix Equation AX + XB = C". In: *Commun. ACM* 15.9 (1972), pp. 820–826. DOI: 10.1145/361573.361582.

[5] Ulrike Baur, Peter Benner, and Lihong Feng. "Model order reduction for linear and nonlinear systems: a system-theoretic perspective". In: *Archives of Computational Methods in Engineering* 21 (2014), pp. 331–358. DOI: 10.1007/s11831-014-9111-2.

[6] Peter Benner and Tobias Breiten. "Low rank methods for a class of generalized Lyapunov equations and related issues". In: *Numerische Mathematik* 124.3 (2013), pp. 441–470. DOI: 10.1007/s00211-013-0521-0.

[7] Peter Benner and Zvonimir Bujanović. "On the solution of large-scale algebraic Riccati equations by using low-dimensional invariant subspaces". In: *Linear Algebra and its Applications* 488 (2016), pp. 430–459. DOI: 10.1016/j.laa.2015.09.027.

[8] Peter Benner, Zvonimir Bujanović, Patrick Kürschner, and Jens Saak. "A Numerical Comparison of Different Solvers for Large-Scale, Continuous-Time Algebraic Riccati Equations and LQR Problems". In: *SIAM Journal on Scientific Computing* 42.2 (2020), A957–A996. DOI: 10.1137/18M1220960.

[9] Peter Benner, Zvonimir Bujanović, Patrick Kürschner, and Jens Saak. "RADI: a low-rank ADI-type algorithm for large scale algebraic Riccati equations". In: *Numerische Mathematik* 138.2 (Feb. 2018), pp. 301–330. DOI: 10.1007/s00211-017-0907-5.

[10] Peter Benner, Albert Cohen, Mario Ohlberger, and Karen Willcox, eds. *Model Reduction and Approximation: Theory and Algorithms*. SIAM, 2017. ISBN: 978-1-611974-81-2.

[11] Peter Benner, Matthias Heinkenschloss, Jens Saak, and Heiko K. Weichelt. "An inexact low-rank Newton–ADI method for large-scale algebraic Riccati equations". In: *Applied Numerical Mathematics* 108 (2016), pp. 125–142. DOI: 10.1016/j.apnum.2016.05.006.

[12] Peter Benner, Patrick Kürschner, and Jens Saak. "An improved numerical method for balanced truncation for symmetric second-order systems". In: *Mathematical and Computer Modelling of Dynamical Systems* 19.6 (2013), pp. 593–615. DOI: 10.1080/13873954.2013.794363.

[13] Peter Benner, Patrick Kürschner, and Jens Saak. "Efficient handling of complex shift parameters in the low-rank Cholesky factor ADI method". In: *Numerical Algorithms* 62.2 (Apr. 2012), pp. 225–251. DOI: 10.1007/s11075-012-9569-7.

[14] Peter Benner, Ren-Cang Li, and Ninoslav Truhar. "On the ADI method for Sylvester equations". In: *Journal of Computational and Applied Mathematics* 233.4 (2009), pp. 1035–1045. DOI: 10.1016/j.cam.2009.08.108.

[15] Peter Benner and Jens Saak. "A Semi-Discretized Heat Transfer Model for Optimal Cooling of Steel Profiles". In: *Dimension Reduction of Large-Scale Systems*. Ed.

by Peter Benner, Danny C. Sorensen, and Volker Mehrmann. Springer Berlin Heidelberg, 2005, pp. 353–356. DOI: 10.1007/3-540-27909-1_19.

[16] Mario Berljafa. "Rational Krylov Decompositions: Theory and Applications". PhD thesis. The University of Manchester, 2017. URL: http://eprints.maths.manchester.ac.uk/id/eprint/2529.

[17] Mario Berljafa, Steven Elsworth, and Stefan Güttel. *A Rational Krylov Toolbox for MATLAB*. Tech. rep. MIMS EPrint 2014.56. Manchester Institute for Mathematical Sciences, The University of Manchester, 2014. URL: http://eprints.maths.manchester.ac.uk/id/eprint/2773.

[18] Mario Berljafa and Stefan Güttel. "Generalized Rational Krylov Decompositions with an Application to Rational Approximation". In: *SIAM Journal on Matrix Analysis and Applications* 36.2 (2015), pp. 894–916. DOI: 10.1137/140998081.

[19] Mario Berljafa and Stefan Güttel. "Parallelization of the Rational Arnoldi Algorithm". In: *SIAM Journal on Scientific Computing* 39.5 (2017), S197–S221. DOI: 10.1137/16M1079178.

[20] Christian Bertram and Heike Faßbender. "A Link Between Gramian-Based Model Order Reduction and Moment Matching". In: *Model Reduction of Complex Dynamical Systems*. Ed. by Peter Benner, Tobias Breiten, Heike Faßbender, Michael Hinze, Tatjana Stykel, and Ralf Zimmermann. Cham: Springer International Publishing, 2021, pp. 119–139. ISBN: 978-3-030-72983-7. DOI: 10.1007/978-3-030-72983-7_6.

[21] Christian Bertram and Heike Faßbender. "A quadrature framework for solving Lyapunov and Sylvester equations". In: *Linear Algebra and its Applications* 622 (2021), pp. 66–103. DOI: https://doi.org/10.1016/j.laa.2021.03.029.

[22] Christian Bertram and Heike Faßbender. "Riccati ADI: Existence, uniqueness and new iterative methods". In: *arXiv e-prints* (2020). arXiv: 2004.11212 [math.NA].

[23] Dario A. Bini, Bruno Iannazzo, and Beatrice Meini. *Numerical Solution of Algebraic Riccati Equations*. Society for Industrial and Applied Mathematics, 2011. ISBN: 978-1-611972-08-5. DOI: 10.1137/1.9781611972092.

[24] John C. Butcher. "Implicit Runge-Kutta Processes". In: *Mathematics of Computation* 18.85 (1964), pp. 50–64. DOI: 10.2307/2003405.

[25] John C. Butcher. *Numerical Methods for Ordinary Differential Equations*. Wiley, 2016. ISBN: 9781119121503. DOI: 10.1002/9781119121534.

[26] John C. Butcher. "On the implementation of implicit Runge-Kutta methods". In: *BIT* 16.3 (Sept. 1976), pp. 237–240. DOI: 10.1007/bf01932265.

[27] Biswa N. Datta. "Linear and numerical linear algebra in control theory: Some research problems". In: *Linear Algebra and its Applications* 197/198 (1994), pp. 755–790. DOI: 10.1016/0024-3795(94)90512-6.

[28] Timothy A. Davis and Yifan Hu. "The University of Florida Sparse Matrix Collection". In: *ACM Trans. Math. Softw.* 38.1 (Dec. 2011). DOI: 10.1145/2049662.2049663.

[29] Vladimir Druskin, Leonid Knizhnerman, and Valeria Simoncini. "Analysis of the Rational Krylov Subspace and ADI Methods for Solving the Lyapunov Equation". In: *SIAM Journal on Numerical Analysis* 49.5 (2011), pp. 1875–1898. DOI: 10.1137/100813257.

[30] Steven Elsworth and Stefan Güttel. "The Block Rational Arnoldi Method". In: *SIAM Journal on Matrix Analysis and Applications* 41.2 (2020), pp. 365–388. DOI: 10.1137/19M1245505.

[31] Roland W. Freund. "Krylov-subspace methods for reduced-order modeling in circuit simulation". In: *Journal of Computational and Applied Mathematics* 123.1 (2000). Numerical Analysis 2000. Vol. III: Linear Algebra, pp. 395–421. DOI: 10.1016/S0377-0427(00)00396-4.

[32] Zoran Gajic and Muhammad T. Qureshi. *Lyapunov Matrix Equation in System Stability and Control*. Math. in Science and Engineering. San Diego, CA: Academic Press, 1995. ISBN: 9780080535678.

[33] Kyle A. Gallivan, Antoine Vandendorpe, and Paul Van Dooren. "Model reduction via truncation: an interpolation point of view". In: *Linear Algebra and its Applications* 375 (2003), pp. 115–134. DOI: 10.1016/S0024-3795(03)00648-7.

[34] G. Golub, S. Nash, and C. Van Loan. "A Hessenberg-Schur method for the problem AX + XB= C". In: *IEEE Transactions on Automatic Control* 24.6 (1979), pp. 909–913. DOI: 10.1109/TAC.1979.1102170.

[35] Gene H. Golub and Charles F. van Loan. *Matrix Computations*. 4th. Johns Hopkins University Press, 2013. ISBN: 978-1-421-40794-4.

[36] Lars Grasedyck. "Existence of a low rank or $\mathcal{H}$-matrix approximant to the solution of a Sylvester equation". In: *Numerical Linear Algebra with Applications* 11.4 (2004), pp. 371–389. DOI: 10.1002/nla.366.

[37] Eric Grimme. "Krylov projection methods for model reduction". PhD thesis. University of Illinois at Urbana-Champaign, May 1997.

[38] Serkan Gugercin, Danny C. Sorensen, and Athanasios C. Antoulas. "A Modified Low-Rank Smith Method for Large-Scale Lyapunov Equations". In: *Numerical Algorithms* 32 (2003), pp. 27–55. DOI: 10.1023/A:1022205420182.

[39] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. 2nd. Springer, 2006. ISBN: 978-3-540-30666-5. DOI: 10.1007/3-540-30666-8.

[40] Ernst Hairer, Syvert Norsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I*. 2nd. Springer, 1993. ISBN: 978-3-540-78862-1. DOI: 10.1007/978-3-540-78862-1.

[41] Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II*. 2nd. Springer, 1996. ISBN: 978-3-642-05221-7. DOI: 10.1007/978-3-642-05221-7.

[42] Mohammed Heyouni and Khalide Jbilou. "An extended block Arnoldi algorithm for large-scale solutions of the continuous-time algebraic Riccati equation". In: *Electronic Transactions on Numerical Analysis* 33 (Jan. 2009), pp. 53–62.

[43] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Matrix Analysis. Cambridge University Press, 2013. ISBN: 9780521839402.

[44] Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991. ISBN: 978-0-511-84037-1. DOI: 10.1017/CBO9780511840371.

[45] Arieh Iserles. *A First Course in the Numerical Analysis of Differential Equations*. 2nd. Cambridge Texts in Applied Mathematics. Cambridge University Press, 2008. ISBN: 978-0-521-73490-5. DOI: 10.1017/CBO9780511995569.

[46] Christopher A. Kennedy and Mark H. Carpenter. *Diagonally Implicit Runge-Kutta Methods for Ordinary Differential Equations. A Review*. Technical Memorandum NASA-TM-2016-219173. Hampton, Virginia 23681-2199: National Aeronautics and Space Administration Langley Research Center, Mar. 2016.

[47] Jan G. Korvink and Evgenii B. Rudnyi. "Oberwolfach Benchmark Collection". In: *Dimension Reduction of Large-Scale Systems*. Ed. by Peter Benner, Danny C. Sorensen, and Volker Mehrmann. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 311–315. ISBN: 978-3-540-27909-9. DOI: 10.1007/3-540-27909-1_11.

[48] Patrick Kürschner. "Efficient Low-Rank Solution of Large-Scale Matrix Equations". PhD thesis. Aachen: Shaker Verlag, 2016. ISBN: 978-3-8440-4385-3.

[49] Sanjay Lall, Jerrold E. Marsden, and Sonja Glavaški. "A subspace approach to balanced truncation for model reduction of nonlinear control systems". In: *International Journal of Robust and Nonlinear Control* 12.6 (2002), pp. 519–535. DOI: 10.1002/rnc.657.

[50] Peter Lancaster and Leiba Rodman. *Algebraic Riccati Equations*. Oxford science publications. Clarendon Press, 1995. ISBN: 9780191591259.

[51] Jing-Rebecca Li and Jacob White. "Low Rank Solution of Lyapunov Equations". In: *SIAM Journal on Matrix Analysis and Applications* 24.1 (2002), pp. 260–280. DOI: 10.1137/S0895479801384937.

[52] Yiding Lin and Valeria Simoncini. "A new subspace iteration method for the algebraic Riccati equation". In: *Numerical Linear Algebra with Applications* 22.1 (2015), pp. 26–47. DOI: 10.1002/nla.1936.

[53] Jens Saak, Martin Köhler, and Peter Benner. *M-M.E.S.S.-2.0 – The Matrix Equations Sparse Solvers library.* see also: www.mpi-magdeburg.mpg.de/projects/mess. Aug. 2019. DOI: 10.5281/zenodo.3368844.

[54] B. C. Moore. "Principal component analysis in linear systems: Controllability, observability, and model reduction". In: *IEEE Trans. Automat. Control* AC-26 (1981), pp. 17–32. DOI: 10.1109/TAC.1981.1102568.

[55] Christian Moosmann, Evgenii B. Rudnyi, Andreas Greiner, and Jan G. Korvink. "Model order reduction for linear convective thermal flow". In: *THERMINIC 2004*. Sophia Antipolis, France, 2004.

[56] Mark R. Opmeer. "Model order reduction by balanced proper orthogonal decomposition and by rational interpolation". In: *IEEE Transactions on Automatic Control* AC-57 (2012), pp. 472–477. DOI: 10.1109/tac.2011.2164018.

[57] Davide Palitta. "The projected Newton-Kleinman method for the algebraic Riccati equation". In: *arXiv e-prints* (Jan. 2019). arXiv: 1901.10199 [math.NA].

[58] D. Peaceman and H. Rachford Jr. "The Numerical Solution of Parabolic and Elliptic Differential Equations". In: *Journal of the Society for Industrial and Applied Mathematics* 3.1 (1955), pp. 28–41. DOI: 10.1137/0103003.

[59] Thilo Penzl. "A Cyclic Low-Rank Smith Method for Large Sparse Lyapunov Equations". In: *SIAM Journal on Scientific Computing* 21.4 (1999), pp. 1401–1418. DOI: 10.1137/S1064827598347666.

[60] Clarence W. Rowley. "Model Reduction for fluids, Using Balanced Proper Orthogonal Decomposition". In: *I. J. Bifurcation and Chaos* 15.3 (2005), pp. 997–1013. DOI: 10.1142/S0218127405012429.

[61] Youcef Saad. "Numerical Solution of Large Lyapunov Equations". In: *Signal Processing, Scattering and Operator Theory, and Numerical Methods, Proc. MTNS-89*. Birkhauser, 1990, pp. 503–511.

[62] Valeria Simoncini. "Analysis of the Rational Krylov Subspace Projection Method for Large-Scale Algebraic Riccati Equations". In: *SIAM Journal on Matrix Analysis and Applications* 37.4 (2016), pp. 1655–1674. DOI: 10.1137/16M1059382.

[63] Valeria Simoncini. "Computational Methods for Linear Matrix Equations". In: *SIAM Review* 58.3 (2016), pp. 377–441. DOI: 10.1137/130912839.

[64] Valeria Simoncini, Daniel B. Szyld, and Marlliny Monsalve. "On two numerical methods for the solution of large-scale algebraic Riccati equations". In: *IMA Journal of Numerical Analysis* 34.3 (Aug. 2013), pp. 904–920. DOI: 10.1093/imanum/drt015.

[65] John R. Singler. "Convergent snapshot algorithms for infinite-dimensional Lyapunov equations". In: *IMA Journal of Numerical Analysis* 31.4 (2011), pp. 1468–1496. DOI: 10.1093/imanum/drq028.

[66] Lloyd N. Trefethen and David Bau III. *Numerical Linear Algebra*. Philadelphia: SIAM, 1997. ISBN: 978-0-898-71361-9.

[67] Karen Willcox and Jaime Peraire. "Balanced model reduction via the proper orthogonal decomposition". In: *AIAA Journal* 40.11 (2002), pp. 2323–2330. DOI: 10.2514/2.1570.

[68] Thomas Wolf. "$\mathcal{H}_2$ pseudo-optimal model order reduction". Dissertation. München: Technische Universität München, 2014. ISBN: 978-3-8439-1926-5.

[69] Thomas Wolf and Heiko K. F. Panzer. "The ADI iteration for Lyapunov equations implicitly performs $\mathcal{H}_2$ pseudo-optimal model order reduction". In: *International Journal of Control* 89.3 (2016), pp. 481–493. DOI: 10.1080/00207179.2015.1081985.

[70] Ngai Wong and Venkataramanan Balakrishnan. "Fast Positive-Real Balanced Truncation Via Quadratic Alternating Direction Implicit Iteration". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 26.9 (Sept. 2007), pp. 1725–1731. DOI: 10.1109/TCAD.2007.895617.

[71] Ngai Wong and Venkataramanan Balakrishnan. "Quadratic alternating direction implicit iteration for the fast solution of algebraic Riccati equations". In: *Proceedings of 2005 International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS 2005.* Vol. 2005. Jan. 2006, pp. 373–376. DOI: 10.1109/ISPACS.2005.1595424.

[72] Zhi-Hua Xiao, Yao-Lin Jiang, and Zhen-Zhong Qi. "Finite-time balanced truncation for linear systems via shifted Legendre polynomials". In: *Systems & Control Letters* 126 (2019), pp. 48–57. DOI: 10.1016/j.sysconle.2019.03.004.